Enterprise Solutions • Data Management • XML Labs

XML JOURNAL

The Ultimate XML Enterprise Resource

July 2002 | Volume: 3 Issue: 7

XML-JOURNAL.COM

web services EDGE world tour 2002

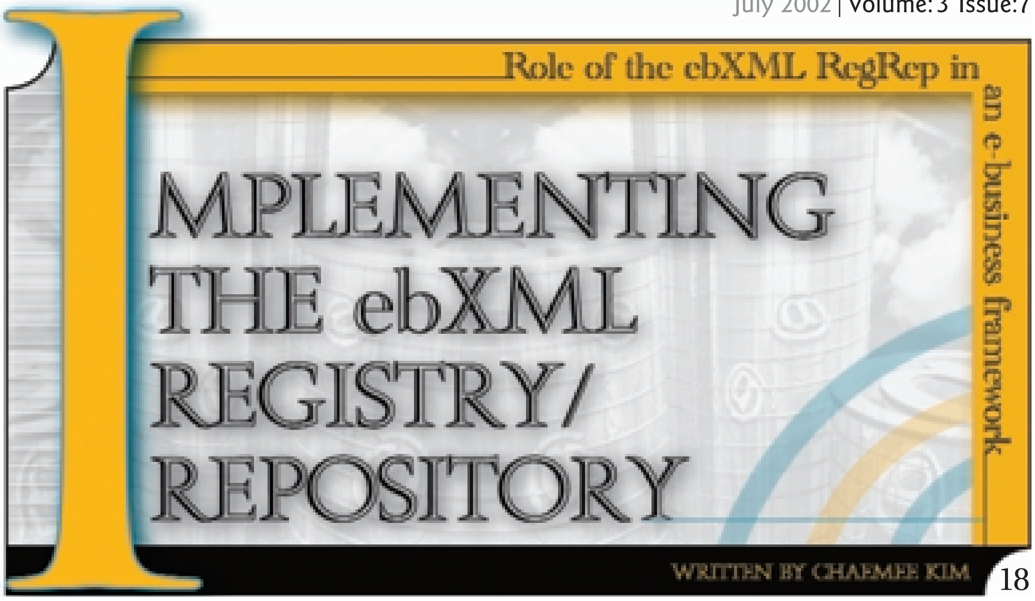
COMING TO A CITY NEAR YOU

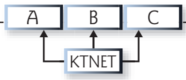
SEE PAGE 36 FOR DETAILS

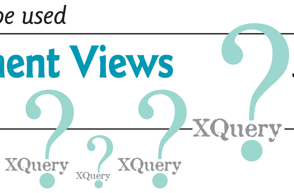
2002

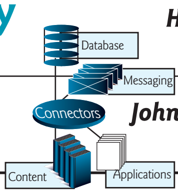
BOSTON	JULY 10
SAN FRANCISCO	AUGUST 6
SEATTLE	AUGUST 27
AUSTIN	SEPTEMBER 10
LOS ANGELES	SEPTEMBER 19
SAN JOSE	OCTOBER 3

AND MANY MORE!


Role of the ebXML RegRep in an e-business framework
IMPLEMENTING THE ebXML REGISTRY/ REPOSITORY
WRITTEN BY CHAEMEE KIM







SYS-CON MEDIA

IBM

www.ibm.com/websphere/winning

IBM

www.ibm.com/websphere/winning

Sprint

www.skc.com

EDITORIAL OFFICES
SYS-CON MEDIA
135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 **FAX:** 201 782-9637
XML-JOURNAL (ISSN# 1534-9780)
is published monthly (12 times a year)
by SYS-CON Publications, Inc.
Periodicals postage pending
Montvale, NJ 07645 and additional mailing offices.
POSTMASTER: Send address changes to:
XML-JOURNAL, SYS-CON Publications, Inc.,
135 Chestnut Ridge Road, Montvale, NJ 07645.

Where Does *XML-J* Fit in the IT Spectrum?

WRITTEN BY **JP MORGENTHAL**

The publisher of *XML-Journal*, SYS-CON Media, is always looking to widen and deepen its coverage of the *i*-technologies that are covered by its world-beating stable of publications. Given SYS-CON's other publications, which include ***Java Developer's Journal*** (the leading Java development publication), ***Web Services Journal***, ***Wireless Business & Technology***, ***ColdFusion Developer's Journal***, ***BEA WebLogic Developer's Journal***, ***WebSphere Developer's Journal***, and ***PowerBuilder Developer's Journal***, XML has always been covered extensively as a topic even before one turns to ***XML-J***, begging the question of whether or not a standalone XML publication is justifiable.

John Evdemon, my coeditor-in-chief, and I were a united voice for keeping the standalone XML publication. However, a focus on developing XML applications arguably has considerable overlap now with the existing base of publications. That said, there is a need in the market for information pertaining to the design, best practices, and business benefit of using XML to solve particular types of problems. Hence there should be a publication that is focused on delivering exactly this type of information.

This states the purpose for a standalone XML publication, but I thought it important to briefly define the role of *XML-Journal* as it relates to the other SYS-CON publications. It's also important to note that *XML-J* is one of the few publications from SYS-CON that doesn't contain the word *Developer* in the title. At the time of its creation, the title was carefully selected to identify the much larger role that XML plays in the information industry. However, as is the course of many technologies, most of our initial readers and authors of articles for the magazine were the technologists that were developing and advancing the state of the technology.

Now, in mid-2002, XML has a much broader audience than developers alone. Indeed, the CEO, CIO, CTO, CFO, vice president, director, and line-of-business manager all understand to some degree that XML will play a role in how their systems interact with other systems and with the outside world. To this end, this community is now reaching out to better understand the impact that XML will have on their companies. They want to know about the successes and failures of those that have been implementing it. They want to know about the skills required to use XML in their organization, and they want to know which standards they should watch and, perhaps, participate in developing.

XML-Journal is perfectly positioned to adapt to these new circumstances. But what does that mean about the content of this publication relative to other SYS-CON publications – for example, *Web Services Journal*?

WSJ is an excellent example to start with to discuss *XML-Journal*'s placement in the spectrum of SYS-CON Media's *i*-technology publications. Clearly, Web services has an XML component to it, but so do Enterprise Resource Planning (ERP) and portals. This doesn't mean that *WSJ* should supersede a publication devoted to the advancement and study of the XML industry.

Web services is a methodology for the development of loosely coupled, late-bound applications using standard Internet technologies. The focus of *Web Services Journal* is on the study and advancement of this methodology. Sure, there will be technical articles in both publications that look at building SOAP and WSDL, but the overall content of the two publications is very well defined.

Readers of SYS-CON's excellent developer journals should expect to find a robust treatment of XML in those publications as well. *Java Developer's Journal* is an excellent resource for articles about processing XML in Java using standards like JAXP, JAXR, and JAXB. *ColdFusion Developer's Journal* is a great resource for ColdFusion developers to find out how to incorporate XML into their server-side applications.

However, if you're looking for a publication that will focus on a programming language- and vendor-independent treatment of XML (the tenets of XML), you'll be certain to find that only in *XML-Journal*.

With XML popping up in the far reaches of computing, it's understandable that confusion will be widespread. Hopefully, this editorial better defines *XML-J*'s purpose and relationship to the other SYS-CON publications. We hope you will enjoy the new format and find it even more interesting and useful than before.

JPM@SYS-CON.COM

ABOUT THE EDITOR

JP Morgenthal is an independent consultant based in northern Virginia. The author of two well-regarded books and numerous articles on XML, EAI, B2B, ERP, XRM, and SCM, JP has served as consultant to Sabre Group, Lockheed-Martin, Citibank, ADP, and Kelloggs, among others.

HiTSoftware

www.hitsw.com/xmlrdb

SYS-CON MEDIA
135 Chestnut Ridge Rd., Montvale, NJ 07645
TELEPHONE: 201 802-3000 **FAX:** 201 782-9637

PUBLISHER, PRESIDENT, and CEO
Fuat A. Kircaali fuat@sys-con.com

BUSINESS DEVELOPMENT
VP, Business Development
Grisha Davida grisha@sys-con.com
Chief Operating Officer
Mark Harabedian mark@sys-con.com

ADVERTISING
Senior VP, Sales & Marketing
Carmen Gonzalez carmen@sys-con.com
VP, Sales & Marketing
Miles Silverman miles@sys-con.com
Advertising Director
Robyn Forma robyn@sys-con.com
Advertising Account Manager
Megan Ring megan@sys-con.com
Associate Sales Managers
Carrie Gebert carrieg@sys-con.com
Kristin Kuhnle kristin@sys-con.com
Alisa Catalano alisa@sys-con.com
Leah Hittman leah@sys-con.com

SYS-CON EVENTS
VP, Events
Cathy Walters cathyw@sys-con.com
Conference Manager
Michael Lynch mike@sys-con.com
Regional Sales Managers, Exhibits
Michael Pesick michael@sys-con.com
Richard Anderson richarda@sys-con.com

CUSTOMER RELATIONS / JDJ STORE
Manager, Customer Relations / JDJ Store
Anthony D. Spitzer tony@sys-con.com
Customer Service Representative
Margie Downs margie@sys-con.com

WEB SERVICES
Webmaster
Robert Diamond robert@sys-con.com
Web Designers
Stephen Kilmurray stephen@sys-con.com
Christopher Croce chris@sys-con.com
Catalin Stancesco catalin@sys-con.com
Online Editor
Lin Goetz lin@sys-con.com

ACCOUNTING
Chief Financial Officer
Bruce Kanner bruce@sys-con.com
Assistant Controller
Judith Calnan judith@sys-con.com
Accounts Receivable
Jan Braidech jan@sys-con.com
Accounts Payable
Joan LaRose joan@sys-con.com
Accounting Clerk
Betty White betty@sys-con.com

SUBSCRIPTIONS
SUBSCRIBE@SYS-CON.COM
1 888 303-5282
For subscriptions and requests for bulk orders,
please send your letters to Subscription Department
Cover Price: $6.99/issue
Domestic: $77.99/yr (12 issues)
Canada/Mexico: $99.99/yr
all other countries $129.99/yr
(U.S. Banks or Money Orders)
Back issues: $12 U.S. $15 all others

RELAX NG: The Power Is in the Patterns

WRITTEN BY **TOM GAVEN**

Schema languages are languages that allow you to specify the structure of XML instance documents. RELAX NG (see www.relaxng.org) is an XML schema language that is considered to be simple, yet powerful. This article gives an overview of an important concept of the RELAX NG schema language called *patterns*. The power of RELAX NG can be found in its patterns.

Schema languages also describe the allowed names of elements and attributes that are found in XML instance documents. And they allow you to specify element ordering, occurrence, and allowed content, like simple text, or datatypes, like integers. Some examples of schema languages are W3C XML Schema, RELAX NG, Schematron, and DTD.

RELAX NG differs from other schema languages in that it's built around the concept of patterns. To understand the power of RELAX NG, you must first understand the basic RELAX NG patterns and how they can be combined.

Let's begin by taking a look at the following XML instance document:

```
<document pages="1"  year="2002" >
   <title>Patterns are fun</title>
   <author>Tom Gaven</author>
   <para>
      This is the <b>first</b> paragraph.
   </para>
   <para>
      And this is the <b>second</b>.
   </para>
   <prints>5 60 45</prints>
</document>
```

This XML instance document contains elements and text. To give you an example of the RELAX NG patterns found in this XML instance document, we could say (ignoring attributes) that the pattern for the *document* element would be element *title*, followed by element *author*, followed by one or more *para* elements. The pattern for the para element would be a mixture of text and element b. The *title, author*, and *b* elements all have a similar pattern, text.

Listing 1 is a complete RELAX NG schema document that shows this pattern syntax. The syntax used in the listing is the compact syntax for RELAX NG. This schema can be used to validate the instance document. Line 1 is a top-level declaration that specifies we're using XML Schema datatypes tied to the "xsd" prefix. The keyword "start" specifies the root element of the instance document. (Datatypes, start, element, attribute, text, and list are all keywords.) RELAX NG's compact syntax uses the same occurrence symbols as DTDs (+,?,*).

The RELAX NG patterns are found to the right of the equals (=) sign. Definition names are to the left. Note that patterns can be combined, like the pattern "pages, year, title, author, para+". The comma specifies an ordering of the patterns. You can also group (or nest) patterns using parentheses. The (text | b)* pattern specifies zero or more sets of text or b elements.

Patterns

Element and attribute patterns

The first two patterns we'll discuss are the patterns for elements and attributes. Following is the (compact) syntax for the element and attribute patterns:

```
P1. element nameClass { pattern }
P2. attribute nameClass { pattern }
```

These two patterns are recursive in that they allow other nested patterns. A nameClass is another great feature of RELAX NG, but for now just think of the nameClass as simply the name of the element or attribute. In the schema document lines 3–10 all use the element or attribute pattern.

Datatype patterns

```
P3. datatypeName params?
P4. datatypeName? DatatypeValue
P5. "text"
```

Pattern P3 is used on lines 4 and 10 in the schema document. It allows you to specify the datatype allowed for the content of an element or attribute.

Pattern P4 is used on line 5. The text "2002" is the DatatypeValue field. This specifies that the year element must have the content "2002". Pattern P5 is the text pattern. The keyword "text" specifies that text content is allowed.

Ref (Reference) patterns

```
P6.  ref
P7.  "(" pattern ")"
```

Pattern P6 allows patterns to reference definitions, which in turn define other patterns. The ref pattern is used in the schema document in Listing 1 in lines 2, 3, and 8. For exam-

Continued on page 8

Reader Feedback

A Pro Approves

I took a look at the June issue [Volume 3, issue 6], and I like the content better than I have liked recent issues of *XML-J*. The new editorial team looks promising!…I will definitely keep a closer eye on the content…

Mike Champion
mc@exegesis.org

Reader Finds Another Way

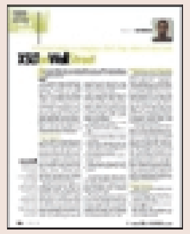
I read Coco Jaenicke's article "Middle-Tier DataManagement" [Vol. 3, issue 4] and found it interesting and useful for our data warehouse project. We use a lot of tools to execute batch programs. Each tool creates various formats of log files, which are viewable only if we open the tool or go to a Unix directory. It occurred to me that we can use the XML technology to "format" the diverse logs of our tools so they can be viewed using the IE browser, or a PDA if we are on the road and want to monitor the logs, for example.

Nice article. More of these.

Pedro Magbitang
via-mail

"XSLT on Wall Street"

Sam Natarajan's piece on using XSLT in the real world [Vol. 3, issue 5] was well written and extremely useful to me. I am embarking on a project where we have decided to use XML in my company and we were going to store all the business rules in Enterprise JavaBeans, over my objections. Thanks to this article I was able to convince my peers that we'd have a lot more flexibility if we put as many business rules as we could into the transformation logic written in XSLT. Good on ya, Sam!



Troy Brentano
via e-mail

Yes, But…

I liked John Ogilvie's treatment of the status of Web services and ebXML ["A Pragmatic Convergence of ebXML and Web Services," Volume 3, issue 4]. His analogy of the formation of the facsimile standard shows he has a good grasp of what it takes to bring a standard to market. However, I find it questionable to say that it's better to have vendors releasing product based on an imperfect standard than waiting for a perfect standard. My experience has shown me that using tools based on imperfect standards often leads to missed deadlines, cost overruns, and a general lack of understanding by nontechnical users of the technology concerning the causes for lack of delivery.

Brad Winslow
via e-mail

Letters may be edited for grammar and clarity as well as length. Please e-mail any comments to John Evedemon (jevdemon@sys-con.com) or JP Morgenthal (jpm@sys-con.com).

Continued from page 7

ple, the "b" reference on the right side of line 8 references the definition (b) on the left side of line 9. Pattern P7 allows patterns to be nested inside parentheses (see line 8 in the schema).

List pattern

```
P8.  list { pattern }
```

The list pattern allows lists of content (separated by white space) to be specified. Line 10 in the schema specifies that the prints element can contain a "list of integers."

More patterns

Overall, 14 different patterns are supported by RELAX NG, eight of which are discussed above. The other available patterns are mixed, empty, notAllowed, grammar, parent ref, and external ref. The first three allow more controls on content; the last three allow for file and grammar modularity.

Power in Combinations

Individually, RELAX NG's patterns are powerful, but you can also combine them to handle more complex validation scenarios.

Combined patterns

```
x = element x { ( list {  xsd:inte-
ger }   |   b ) }
```

```
b = element b { text }
```

Element x contains either a list of integers or an element b.

```
x = element x { "1.0" | "NaN" |  num  }
num = element num { text }
```

Element x contains either the string "1.0", the string "NaN", or a num element.

```
x = element x    { (a,b)|(a,c) }
a = element a    { text }
b = attribute b  { text }
c = attribute c  { text }
```

Element x contains (element a and attribute b) *OR* (element a and attribute c).

RELAX NG patterns are quite powerful, yet easy to learn and use. With their ability to combine, they offer capability not found in either DTDs or XML Schema.

LISTING 1 RELAX NG schema document

```
1   datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"
2   start     = document
3   document = element document { pages
                                , year
                                , title
                                , author
                                , para+
                                , prints }
4   pages     = attribute pages  { xsd:integer }
5   year      = attribute year   { "2002" }
6   title     = element title    { text }
7   author    = element author   { text }
8   para      = element para     { ( text | b )* }
9   b         = element b        { text }
10 prints     = element prints   { list { xsd:integer+ }
}
```

Download the Code
www.sys-con.com/xml

AUTHOR BIO

Tom Gaven has authored over 30 courses in many different technologies, including Assembler, C, C++, Java, OS/2, and Windows. He also authored MindQ's Developer Training for Java program. In the past two years he has architected and developed products with XML, XSLT, XML Schema, RELAX NG, Java, and Schematron. Tom is currently working on tools and courseware to make XML easier to use.

TGAVEN@XMLDISTILLED.COM

Macromedia

www.macromedia.com/go/cfmxad

WRITTEN BY **SHANNON MA**

Model Your Business Processes Using BPML

Just what is BPML, and how does it work?

This article introduces the Business Process Modeling Language, a standard specification to model business processes. To illustrate the elements of BPML, I'll use a simple example that handles purchase orders.

BPML is an XML metalanguage for business process modeling. A standard published by Business Process Management Initiative (BPMI), it provides a universal way to describe business processes in organizations. Business processes modeled using BPML can be deployed and executed on any BPML-compliant Business Process Management System (BPMS).

BPML Fundamentals

Process

Process is the core concept of BPML. Key components of a process include:

- Execution of a set of activities
- Interaction with participants by exchanging messages
- Predefined rules to determine the flow of the execution
- Schedules to govern the start and completion of the execution
- Processing of data by holding the internal data that can be accessed by all its activities

I'll use a simple handlePurchaseOrder process to illustrate how to model a process using BPML. The process contains the following steps:

1. At startup the process will wait for the message "purchase order".
2. When it receives a purchase order, the process will check whether it's an existing customer, and will create a customer account if it's a new customer.
3. The process checks the inventory system to see whether there's enough inventory to fulfill the order.
4. The process sends a shipping bid to all the carriers it contracts. It sets a predefined price, and will accept the first bid that meets the price criterion.
5. The process schedules an order delivery with the selected carrier.
6. The process sends a shipping notification to the customer.

First, we need to define our process:

```
<process name="handlePurchaseOrder">
…
</process>
```

We'll keep adding elements as we go on.

Activities

In BPML there are three types of activities: simple, complex, and process. A process can have one and only one top-level activity.

A *simple* activity is an atomic action that contains no other activity. It performs a task that involves communication with participants. Following is a list of simple activities defined in BPML:

- ***Consume:*** An activity that waits and consumes a message. It will wait until a suitable message is available.
- ***Produce:*** An activity that produces and sends a message to a participant.
- ***Operation:*** A synchronous request/response interaction that exchanges messages with a participant. It has two types, invoking and implementing. *Invoking* produces an outgoing message (a request) to a participant and consumes an incoming message (a response). *Implementing* consumes an incoming message (a request) and produces an outgoing message (a response).
- ***Empty:*** An activity used to describe an action that requires human interaction, such as setting up a meeting or obtaining a signature from a manager.
- ***Exception:*** An activity used to indicate that an error occurs during the execution. Every exception has an exception code.

Next, I'm going to add a few simple activities to our process:

In Step 1 we'll define a Consume that waits for incoming orders with a message named "purchaseOrder".

```
<consume name="waitingForOrder">
 <input message="purchaseOrder"/>
</consume>
```

In Step 3 we'll define an Operation (invoking) that sends out a checking inventory request to the inventory system and waits for the checking inventory response. This is an invoking operation in that the output appears before the input. The order of input and output appearing in an operation is how we differentiate invoking and implementing.

```
<operation>
 <participant select=
  "inventorySystem"/>
 <output message=
  "checkInventoryRequest"/>
 <input message=
  "checkInventoryResponse"/>
</operation>
```

AUTHOR BIO
Shannon Ma is a senior software developer specializing in J2EE and XML technology. He is currently working at DCH Systems, a software company that provides a comprehensive Business Process Management System.

Sonic Software

www.sonicsoftware.com/websj

In Step 4 we'll define a Produce that sends out a ship request to all the carriers and a Consume that waits for carriers to send their responses.

```
<produce name="sendingBidToCarriers">
 <participant select="carriers"/>
 <output message="shippingRequest"/>
</produce>
<consume name="waitingForCarrier">
 <input message="shippingResponse"/>
</consume>
```

In Step 5 we'll define a Produce that sends out an order delivery schedule to the selected carrier.

```
<produce ="sendingShippingOrder">
 <participant select=
  "shippingResponse/carrier"/>
 <output message="shippingOrder"/>
</produce>
```

In Step 6 we'll define a Produce that sends out a shipping notification to the customer.

```
<produce ="sendingNotification">
 <participant select=
  "purchaseOrder/customer"/>
 <output message=
  "customerNotification"/>
</produce>
```

A *complex* activity is used to composite one or more activities. Different complex activities provide different flows of executions. A complex activity completes when all its containing activities have completed. Following is a list of complex activities defined in BPML:

- ***Sequence:*** Contains a list of activities that are executed in sequence.
- ***All:*** Contains a list of activities that can be executed in parallel.
- ***Switch:*** Contains a list of activities of which zero or more will be executed, depending on the condition or the rule defined for each individual activity.
- ***Choice:*** Contains a list of paths, each with a set of activities. Only one of the paths will be executed, depending on the incoming message.
- ***Foreach:*** Contains a list of activities that are repeated with different sets of process data.

For our handlePurchaseOrder process we'll define a Sequence and put all the activities to execute in the Sequence. This will be the top-level activity for our process. Within the Sequence we'll put sending delivery schedule and sending customer notification in an All because they're independent and can be executed in parallel. We'll use a Switch for Step 2 of the process to create a new customer account in the Customer Account Database if it is a new customer (see Listing 1).

The third type of activity is *process* activity, which affects only process and process data, but doesn't interact with any participant. Following is a list of process activities defined in BPML:

- ***Spawn:*** Spawns a nested child process under the current process
- ***Join:*** Waits for child process to finish
- ***Complete:*** Causes the current process to complete immediately
- ***Repeat:*** Repeats an activity that has executed before
- ***Assign:*** Transforms data between messages and process data
- ***Release:*** Removes data from the process data

Participant

Any entity that a process communicates is defined as a *participant*. Participants can be business applications (e.g., ERP, CRM), customers, partners, and other processes. They can be either static or dynamic. *Static* participant is defined in the process and is referenced by activities in the process. *Dynamic* participant is retrieved from process data using XPath expression. In our example we define inventorySystem and customerAccountDatabase as static participants; they are referenced by Operation checkingInventory and createNewAccount, respectively. In Produce sendingShippingOrder the participant is the carrier retrieved from shippingResponse. In Produce sendingNotification the participant is the customer retrieved from purchaseOrder.

Message

Messages are used to exchange information between a process and its participants. All messages are in XML format, and a schema is used to define the format. Upon receiving a message, an activity can use rules to decide whether it will consume the message. Consumed messages are transformed into process data using assignment. XPath is used to retrieve and transform messages.

Listing 2 is the XML schema for the message "purchaseOrder". We need to define the schema for every message we use.

Here is the assignment that retrieves the product information portion from the purchaseOrder message and transforms it into a new process data element pendingOrder:

```
<assign target="pendingOrder"
 select="purchaseOrder/product"/>
</assign>
```

Rules can be set to decide whether to consume a particular message. In our example assume that we'll accept the first carrier whose bid is less than $50.

```
<consume ="waitingForCarrier" >
 <input message="shippingResponse">
  <rule condition=
   "shippingResponse/charge <= 50"/>
 </input>
</consume>
```

Scheduling

Schedule is used to set the start time and time limit of an activity. Schedule can be an absolute time or a relative time that references the start or the end of another activity. For our Consume "waitingForCarrier" we set a time limit of 6 hours so that the bid will stop in 6 hours. If there is no bid less than $50 when time expires, a bpml:timeout exception will be thrown.

```
<consume ="waitingForCarrier" >
 <completeBy duration="PT6H"/>
 <input message="shippingResponse">
  <rule condition=
   "shippingResponse/charge <= 50"/>
 </input>
</consume>
```

PT6H is an XPath expression that represents a duration of 6 hours.

Exception

In BPML exceptions can be handled for complex activities through the use of onException. One onException can be defined to cover all exceptions. In this case there will be no exception code associated with the onException. You can also define multiple onExceptions to handle different types of exceptions by specifying exception codes for each onException.

For our Consume "waitingForBid" we set a time limit; hence, a bpml:timeout exception will occur if there is no acceptable bid. To handle the exception, we define an onException that will send an alert message to the manager (see Listing 3).

Transaction

Transaction is an indivisible unit of work that comprises several operations, all or none of which must be performed to preserve integrity. There are two transaction models in BPML, Coordinated and Extended. *Coordinated* transaction is the ACID transaction in which activities within the transaction must either all complete or all abort; it's used for short-lived activities. *Extended* transaction is the Saga transaction in which all activities will complete – in the event of failure, compensating activities will be executed;

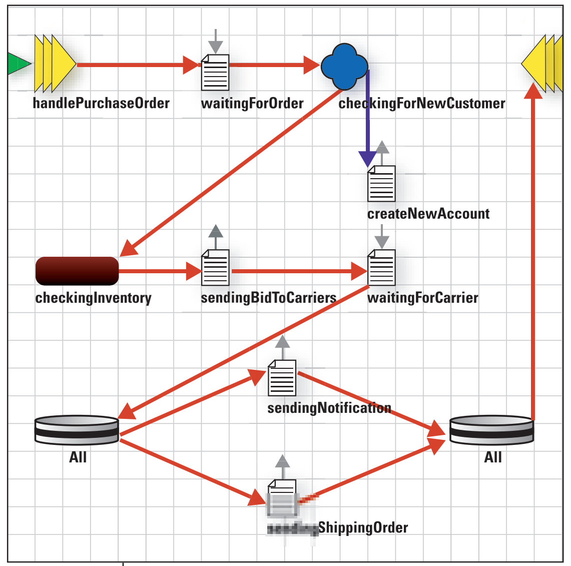


FIGURE 1 Sample for process

it's used for long-lived activities. Both models support transaction types – supported, required, new, nested, and none.

Now let's define a transaction for our All activity so that sending shipping order and sending customer notification will complete or both will abort (see Listing 4).

Figure 1 is a screenshot of our sample process modeled using DCH Systems' BPMS.

BPML and BPMS

BPMS is a system that abstracts and executes business processes. Usually a BPMS provides the following features:

- ***Modeling:*** Provides a visual tool (GUI) for business process modeling
- ***Automation:*** Executes processes in an automated fashion
- ***Monitoring:*** Provides runtime data and statistics about process execution

A BPML-compliant BPMS is a system that models processes in BPML and can import a BPML document and execute with minimal configuration.

• • •

This article introduces BPML and its core concepts. You've seen from the example how to use BPML components to model business processes.

References

This article is based on the BPML working draft 4.0 dated March 8, 2001. The draft was being revised at the time this article was written.

- *For more information on BMPL:* www.bpmi.org
- *For more information on XPath:* www.w3.org/TR/xpath

SHANNON.MA@DCH.COM

LISTING 1

```
<process name="handlePurchaseOrder">

 <sequence name="handlePurchaseOrder">

  <consume name="waitingForOrder">
   <input message="purchaseOrder"/>
  </consume>

  <switch name="checkingForNewCustomer"/>
   <case condition="isNewCustomer() == true"> (remove the
    "/" before ">")
    <produce name="createNewAccount">
     <participant select="customerAccountDatabase"/>
     <output message="accountInfo"/>
    </produce>
   </case>
  </switch>

  <operation name="checkingInventory">
   <participant select="inventorySystem"/>
   <output message="checkInventoryRequest"/>
   <input message="checkInventoryResponse"/>
  </operation>

  <produce name="sendingBidToCarriers">
   <participant select="carriers"/>
   <output message="shippingRequest"/>
  </produce>

  <consume name="waitingForCarrier">
   <input message="shippingResponse"/>
  </consume>

  <all>
   <produce ="sendingShippingOrder">
    <participant select="shippingResponse/carrier"/>
    <output message="shippingOrder"/>
   </produce>
   <produce ="sendingNotification">
    <participant select="purchaseOrder/customer"/>
    <output message="customerNotification"/>
   </produce>
  </all>
 </sequence>

</process>
```

LISTING 2

```
<message name="purchaseOrder" type="request">

            <xs:complexType name="PurchaseOrder">
                          <xs:sequence>
                          <xs:element name="product" type=
                           "Product"></xs:element>
                          <xs:element name="customer" type=
                           "Customer"></xs:element>
                        </xs:sequence>
             </xs:complexType>

             <xs:complexType name="Product">
                     <xs:sequence>
                          <xs:element name="item" type=
                           "xs:string" ></xs:element>
                          <xs:element name="quantity" type=
                           "xs:number" ></xs:element>
                          <xs:element name="shippingAddress"
                           type="xs:string" ></xs:element>
                          <xs:element name="shippingDate" type=
                           "xs:date"></xs:element>
                 </xs:sequence>
            </xs:complexType>
            <xs:complexType name="Customer">
                 <xs:sequence>
                          <xs:element name="name" type=
                           "xs:string" ></xs:element>
                          <xs:element name="address" type=
                           "xs:string" ></xs:element>
                          <xs:element name="phone" type=
                           "xs:string" ></xs:element>
               </xs:sequence>
             </xs:complexType>
</message>
```

LISTING 3

```
<consume ="waitingForCarrier" >
 <completeBy duration="PT6H"/>

 <input message="shippingResponse">
  <rule condition="shippingResponse/charge <= 50"/>
 </input>

 <onException code="bpml:timeout"/>
  <produce name="sendingAlert">
   <participant select="manager"/>
   <output message="alert"/>
  </produce>
 </onException>
</consume>
```

LISTING 4

```
 <transaction model="coordinated" type="new"/>

 <produce ="sendingShippingOrder">
  <participant select="shippingBid/carrier"/>
  <output message="shippingOrder"/>
 </produce>

 <produce ="sendingNotification">
  <participant select=" purchaseOrder/customer"/>
  <output message="customerNotification"/>
 </produce>
</all>
```

Download the Code
www.sys-con.com/xml

Muscular Dystrophy Association

www.als.mdausa.org

WRITTEN BY **J. RHETT AULTMAN**

Instant Messaging with Jabber

An open standard protocol for IM

Of all of the possible technologies to arise out of the first consumer Internet revolution, instant messaging (IM) is a bit of an enigma. Its incredible popularity challenges that of the Internet's first "killer app," e-mail, yet it does so mostly by offering the features of e-mail in a slightly different package.

At the core of any IM system lies the capacity to rapidly send a message from one client to another. The second most important feature in an IM system is the capacity to recognize when certain users of the system have become available. When these two features are put together, it becomes possible to send messages rapidly to a target and be reasonably certain that the target is available to receive them. In contrast, with e-mail there is no guarantee of when, if ever, the target will choose to receive a message. Providing many of the desirable features of wireless paging and e-mail, IM would seem a silver bullet in an enterprise's messaging infrastructure.

Three important hurdles have impeded the progress of enterprise IM: lack of standards, security, and the ability to readily integrate enterprise applications into an IM framework.

Knocking Down the Hurdles

As has often been the case with computing infrastructure over the last decade, the development community has come together to generate an open standard protocol for IM. The name of that protocol, appropriately enough, is *Jabber*. A nearly complete change from the proprietary IM protocols of companies like ICQ, AOL, and Yahoo, Jabber's unique design strategy is well designed for use in the enterprise world. Here are the reasons:

- ***Jabber is an open standard:*** The DTDs defining the Jabber protocol are available to anyone wishing to read them, allowing anyone access to an understanding of the protocol. The Jabber Software Foundation has recently applied to the ITEF to have Jabber considered for an Informational RFC, which provides additional public access to the protocol.
- ***Jabber is an XML protocol:*** Not only does this ensure that the Jabber protocol is platform independent, it also facilitates the rapid development of both client and server technology. An XML-based protocol has a leg up on other protocols in that there's no need to parse a data stream for unique tokens – all that's needed is an XML parser. This has caused a wide proliferation of Jabber API libraries in languages like Java, Perl, and .NET.
- ***Much like SOAP, Jabber is agnostic about underlying protocols used to transport it:*** Jabber will function well via any transport that provides data in a stream. The current preferred way to do this is via plain TCP/IP sockets, but SSL sockets and other stream techniques are becoming popular.
- ***Jabber has provisions for security available:*** Not only does Jabber allow for transport-level encryption, but the Jabber protocol also supports digital signatures and encryption using PGP (Pretty Good Privacy) and can protect passwords using SHA1 (Secure Hash Algorithm v. 1.0).

Of all of the features listed, possibly the most important of these is the fact that Jabber is an XML protocol. This carries with it the benefits so common to XML, though it's the extensibility available in Jabber that makes it so successful. The Jabber protocol is rich in features implemented in unique namespaces, but there's no requirement that these "extension namespaces" be implemented by a client. In fact, there are only three simple tag structures that must be handled by a client to have a basic session with the server, though an implementation of a few extension namespaces can give a client some important bells and whistles.

Even more important than Jabber's liberal approach to protocol conformance is the fact that a session with a Jabber server is essentially the exchange of two XML documents: one of the client's data sent to the server, and one of the server's data sent to the client. The transformability of XML gives developers an advantage in converting XML documents to Jabber as well as packets in the Jabber protocol into other IM protocols. Jabber "transports" can give clients the ability to freely communicate over multiple IM networks at the same time. The XML basis for Jabber also allows developers to start with the Jabber protocol as the basis for their own messaging solution and then extend it to accommodate their specific needs.

Developers Flout Tradition

Developers within the Jabber community are also offering an interesting twist on traditional IM concepts – developing automated information services that act like ordinary clients. Multiple methods exist for integrating applications into a Jabber IM network, including Jabber server-side components and the increasingly popular "bots." The applications available range from simple Perl scripts that rotate Web page graphics to Jogger (http://jogger.jabber.org), a Web journal that accepts entries via Jabber messages.

Jabber provides everything needed for a wide array of possible IM activities

How It All Works

All communication in Jabber is broken down into discrete chunks called *packets*. Each packet is an XML element. The core protocol of Jabber is broken down into three packet types: Message, Presence, and a "utility type" known as Info/Query (IQ). The first two of these packet types are fairly intuitive as they're responsible for implementing the two fundamental features of IM. IQ packets provide the final piece of the puzzle by allowing client programs to communicate to the server and each other via a series of "get" and "set" operations. The structure of each query is left to extension namespaces, making IQ an effective wrapper for every action from a client's authentication to the management of the client's roster (also called a "buddy list") to negotiation of file transfers and the Jabber implementation of XML-RPC. The three packet types are defined as children of a single element called the Stream element, which also serves as the root element of the entire session. When the Stream element closes, the document has been completed and the session dies.

Putting it all together, Jabber provides everything needed for a wide array of possible IM activities. Here's how the three different packet types working together allow a client to interact with the world.

After connecting a socket to the Jabber server, the client begins the session by opening the stream:

```
<?xml version="1.0" encoding=
 "UTF-8" ?>
<stream:stream
to="jabber.org"
xmlns="jabber:client"
xmlns:stream=
 "http://etherx.jabber.org/streams">
```

The server, in recognition, sends back a similar response, allowing the server/client XML stream to begin. Now the client must authenticate itself with the server:

```
<iq type="set" id="1"><query xmlns=
 "jabber:iq:auth"><username>foobar
 </username> <password>murple
 </password><resource>Winjab
 </resource></query></iq>
```

Notice the ID attribute in this packet. This attribute is useful in synchronizing server responses to client packets, which may be (and often are) asynchronous. If the authentication credentials are valid, the server sends back a response confirming the authentication:

```
<iq type="result" id="1"/>
```

If the authorization failed, the server would send back an IQ of type "error" and additional information explaining the reason for the error. Errors are generally reported using an error code standard similar to that of HTTP.

The client is now ready to begin performing IM operations by announcing its presence to the server, which broadcasts the statement of availability to all interested parties:

```
<presence type="available" id="2"/>
```

At this point the client is visible as being "available" on the system, and it's free to do as it (or its user) wishes. The finer points of what a client can do on Jabber is a topic that's well beyond the scope of this article, but I'll briefly illustrate sending a message.

Let's say that this user (foobar@jabber.org) wanted to send a message to an old friend of his (duncan@jabber.org). His client would formulate the following packet:

```
<message id="3" to=
 "duncan@jabber.org"><subject>Dinner
 </subject><body>Don't forget we're
 having the XML-Journal editors over
 for dinner tonight!</body></message>
```

Once done with his Jabber session, the user of the client would opt to disconnect, and the client would gracefully bow out by ending its stream to the server:

```
</stream:stream>
```

Upon seeing the stream ended by the terminating tag, the server likewise ends the client's stream and notifies interested parties that the client has become unavailable. Both sides then close their sockets.

Getting a Feel for It

Probably the best way to get a feel for Jabber is to be a user. JabberCentral (www.jabbercentral.com) maintains an extensive list of Jabber clients. Just grab an appealing one, use it to register an account on a free server such as Jabber.org, and become acquainted with what's out there in the world of Jabber. Those considering becoming client developers should also explore what API libraries are at their disposal. The Jabber project's Web site (www.jabber.org), as well as JabberCentral, contains links to many API libraries.

Starting your own Jabber IM system at home or work is easy. JabberCentral lists several vendors that are offering Jabber servers for all major platforms. There is also an open source Jabber server available for download at the Jabber project's Web site. It comes with installation documentation, and is designed to be easily built on UNIX operating systems. While each Jabber server implementation has its own installation process, one thing is nearly universal: installation of any Jabber server is quite easy. The Jabber server available from the Jabber project homepage is by far the most hands-on, and after installing a build of the server, the only necessary configuration step is to specify some host name information in the jabber.xml configuration file.

The time is definitely here for Jabber to begin taking its place alongside other open standard XML protocols like SOAP. Next time you consider a messaging solution, consider what Jabber may hold in store for you.

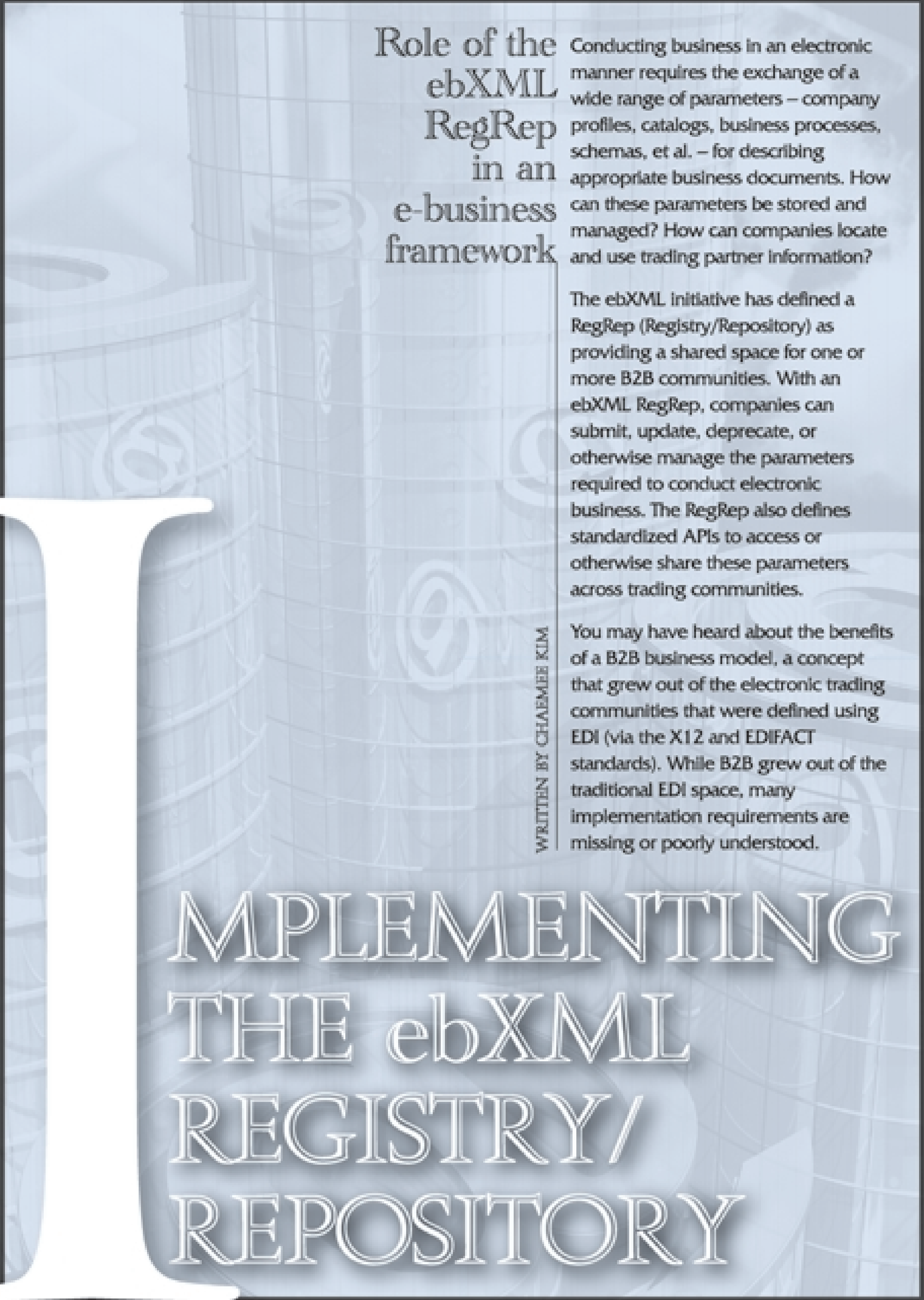
WLIGHT@WEATHERLIGHT.COM

AUTHOR BIO

Rhett Aultman has developed in Java and XML for the past three years. His primary topics of interest are messaging, online services, and distributed systems infrastructures. He currently serves as the lead software developer for Weatherlight Technologies (www.weatherlight.com), an IT consulting co-op based in Tampa, FL.

JABBER ON ITS WAY

Since its inception in 1998, Jabber has been making the slow push toward providing a complete open standard for IM. With the backing of IBM and Hitachi and with three books regarding Jabber development in print, the community is beginning to rally behind Jabber as a standard for IM in much the same fashion as the community has rallied behind HTTP, XML, and SOAP. Jabber is definitely here to stay, and its popularity is on the rise precisely because it effectively tackles the issue of standardization. Commercial Jabber servers, each offering unique features like wireless gateways, have emerged to serve enterprise messaging solutions markets.



IMPLEMENTING THE ebXML REGISTRY/ REPOSITORY

Role of the ebXML RegRep in an e-business framework

WRITTEN BY CHAEMEE KIM

Conducting business in an electronic manner requires the exchange of a wide range of parameters – company profiles, catalogs, business processes, schemas, et al. – for describing appropriate business documents. How can these parameters be stored and managed? How can companies locate and use trading partner information?

The ebXML initiative has defined a RegRep (Registry/Repository) as providing a shared space for one or more B2B communities. With an ebXML RegRep, companies can submit, update, deprecate, or otherwise manage the parameters required to conduct electronic business. The RegRep also defines standardized APIs to access or otherwise share these parameters across trading communities.

You may have heard about the benefits of a B2B business model, a concept that grew out of the electronic trading communities that were defined using EDI (via the X12 and EDIFACT standards). While B2B grew out of the traditional EDI space, many implementation requirements are missing or poorly understood.

What are these missing elements? The traditional B2B business model (again, based on the EDI model) assumes that trading partners have advance knowledge of each other's e-business environments, trading protocols, and procedures. The "discovery" phase is traditionally done offline via a manual process (phone calls, legal contracts, etc.). This approach limits companies to conducting business with a relatively small community of well-known trading partners. A well-defined discovery process would enable most companies to significantly expand the size of their trading communities. The current B2B model, however, doesn't support such a process, forcing trading partner configuration to be accomplished offline.

The ebXML RegRep is designed to close some of the gaps in traditional B2B business models, as B2B alone isn't enough to establish true collaborative commerce. B2B with an ebXML RegRep provides a more advanced B2B model that we call *Business-RegRep-Business*, or *BRB*. ebXML RegRep enables trade parameters to be shared among business peers, and helps to build more dynamic B2B environments based on the discovery and execution of trade agreements with ebXML-enabled trading partners.

Understanding ebXML's BRB Model

ebXML's BRB model can be clarified using a simple formula:

```
BRB = B2R + B2B
```

Trading partners perform a *Business-to-RegRep*, or *B2R*, transaction before conducting B2B transactions. ebXML provides a more dynamic mechanism for finding and conducting business with new trading partners.

The U.S. General Accounting Office recently filed a report, "Electronic Government – Challenges to Effective Adoption of the Extensible Markup Language," with the Senate Committee on Governmental Affairs (view it at www.gao.gov/new.items/d02327.pdf). In the report the GAO describes the current status of the standards for XML registries and raises some concerns regarding the maturity of XML-related technologies as well as the lack of an all-encompassing XML vocabulary. While the report would appear to be a harsh criticism of XML, it draws some exciting conclusions regarding XML-based registries.

According to the report, XML registries should be used to share information between distributed agencies in order to conduct electronic business, effectively implementing an "e-gov" program. As described by the GAO, XML registries will play a very important role in e-government as well as e-business. XML makes integration easy, and an XML registry lowers the barrier for sharing trading parameters.

Note, however, that the GAO report doesn't specifically address the use of an ebXML-compatible RegRep. The report refers to an "XML registry" without regard to its compatibility with ebXML or UDDI. The need for an XML registry in an e-government initiative reflects many of the same needs within private industry. If the ebXML RegRep Specification fulfills the requirements for an e-government initiative, the barriers to ebXML RegRep adoption by various industrial consortia will also be lowered.

ebXML Components and Relationships

ebXML provides a loosely coupled, highly modular framework for conducting e-business. On the other hand, each module is closely related to other modules (although they can also be used independently of one another). From a software perspective this means that ebXML components have high cohesion and low coupling. ebXML Working Groups have been formed to define each module, so the maturity of each component is also different. Each ebXML Working Group develops its specification according to the group's plan, and its progress depends on member participation. This means that the version numbers of each ebXML specification may be quite different. Some are up to version 2.0, while others remain at version 1.1 or 1.2.

ebXML RegRep clients can access the repository via one of two possible interfaces:

1. ***Simple Object Access Protocol:*** SOAP uses simple remote procedure calls (RPC) to send and receive XML messages. The XML messages are used to invoke a Web service and return the results of the invocation.
2. ***ebXML Message Handler Service:*** ebXML MHS is an extension of SOAP and adds functionality such as guaranteed messaging, multihop delivery, and enhanced security.

Relationship between ebXML RegRep and messaging components

ebXML MHS refers to CPA (Collaborating Protocol Agreement) for extracting useful configuration information. CPA refers to a business process specification schema for sharing a business process with trading partners. A business process needs business documents to perform business transactions. Each business document format is defined by reusing the data items in core components.

The ebXML messaging component is the most mature module in the entire ebXML framework. Organizations such as Covisint (Auto Exchange), OAG (a horizontally oriented XML initiative), PAA (Pan-Asian Alliance – an international e-business consortium), RosettaNet (XML initiatives for the electronics industry), and STAR (Standards for Technology in Automotive Retail) have already adopted the ebXML messaging plan to provide support for other ebXML modules in the near future.

The Evolution of ebXML

The most important decision in adopting ebXML is to align your adoption steps with the components of the ebXML framework you need. An example of the approach is shown in Figure 1. This alignment is required because the maturity of ebXML components may affect each adoption step. There are no organizations that can/will adopt the entire ebXML framework at the same time. A phased approach to ebXML adoption will guide you through the shortest path to successfully adopt and implement ebXML.

The most accessible component of the ebXML framework is the ebXML messaging service (sometimes referred to as *TRP* for Transport, Routing, and Packaging). Although the service uses CPA information, the first generation of ebXML messaging can also use an ad hoc configuration file instead of CPA (since the CPA spec is not yet completed). The ebXML messaging service payload (i.e., the "message") can use any standard business document (usually in an XML syntax). The GAO report warns that a wide variety of business document formats and vocabularies may cause barriers to interoperability. While a single, global XML standard for business vocabularies has yet to emerge, several non-XML standards (such as EDI's UN/EDIFACT and X.12) can provide guidelines for defining and processing XML-based transactions.

For the next step of ebXML adoption, a standard protocol profile should be adopted in agreement with CPP (Collaborating Protocol Profile) and CPA. For discovery and negotiation, each trading partner creates a CPP and submits it to an ebXML RegRep. When a CPP is discovered, the potential trading partners exchange CPAs and, once approved, can enter into trade agreements and begin conducting business electronically (B2B). While this scenario will be possible with future versions of the ebXML framework, the current (non-ebXML) B2B model doesn't support such dynamic agreements. The reality of B2B is that the transaction occurs between fixed trading partners. Companies may decide to implement ebXML messaging services while ignoring the entire discovery process. With this approach, potential trading partners simply exchange CPPs and CPAs via e-mail. An ebXML RegRep may be used in a later generation of ebXML adoption, providing support for the dynamic "discovery" process described earlier.

For the third generation of ebXML adoption, companies will adopt ebXML-compatible business processes. These processes may depend on the progress and overall acceptance of collaborative commerce. The automation of business processes for collaboration is the final goal of ebXML and e-business. The focus of ebXML business processes is to provide public processes for collaboration, not a company-specific private process for internal workflow. Companies may face significant challenges integrating their public and private processes.

For the fourth generation of ebXML adoption, companies will focus on reuse, leveraging prebuilt core components to construct standard business documents. The ebXML core components provide a standard dictionary for constructing e-business vocabularies. The ebXML core components data dictionary is designed to be stored within an ebXML registry.

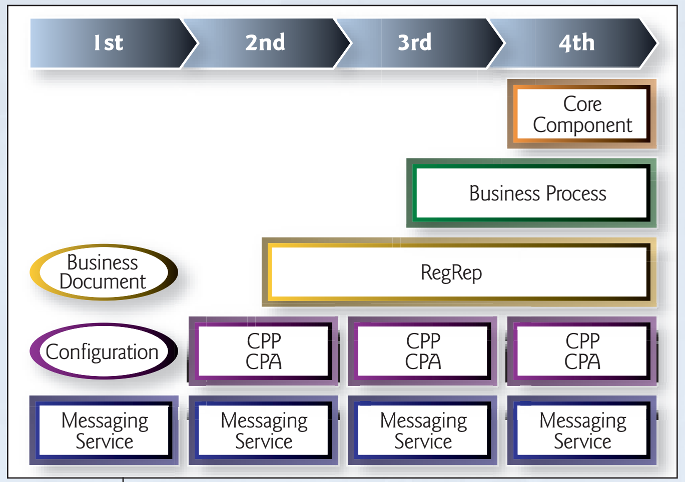


FIGURE 1 | Steps toward ebXML adoption

	UDDI	ebXML RegRep
Registry & Repository	Only Registry	Registry & Repository
Purpose & Scope	Web Services	Global e-Business Part of ebXML Framework Content Management
Actor	Service Provider Service Registry Service Requestor	Registry Guest Content Owner Registry Administrator
Relationship	Using tModel, UDDI specifies ebXML Registry Service	RIM 2.0 will support Web Service Interface - Service, ServiceBinding, SpecificationLink
Organization	IBM, MS, HP, Ariba	UN/CFFACT, OASIS

TABLE 1 | Comparing UDDI and the ebXML RegRep

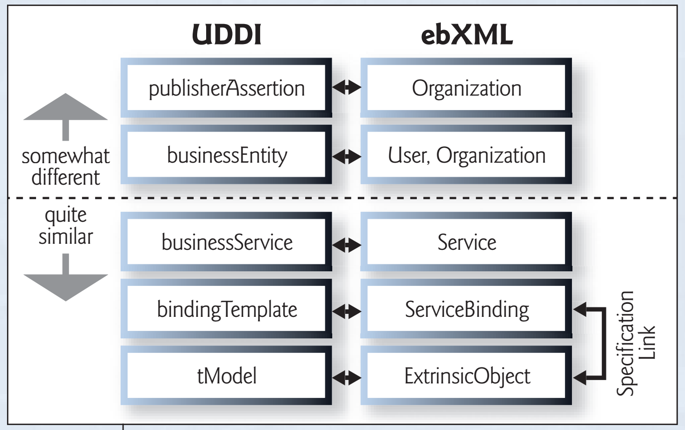


FIGURE 2 | Comparison of structures and entities used by UDDI and ebXML RIM 2.0 vs UDDI

Differences Between ebXML RegRep 2.0 and UDDI

What distinguishes ebXML RegRep from UDDI in Web services? This is one of the questions most frequently asked by companies interested in both ebXML and Web services. ebXML RegRep and UDDI repository each have a different scope and purpose.

An ebXML RegRep, which is more likely to be focused on content management, was designed to store and manage a wide range of electronic trading parameters. UDDI on the other hand was designed to manage the registry metadata associated with a Web service. In short, ebXML RegRep is the registry for B2B, while UDDI is the registry for Web services.

From a UDDI perspective, the UDDI initiative can provide a loosely coupled connection model with an ebXML RegRep. An ebXML RegRep's registry service specification can be published as a SOAP interface, enabling a traditional Web service that describes how to access the ebXML RegRep using a SOAP client.

From an ebXML RegRep perspective, the RegRep can include a Web services registry within an ebXML Registry Information Model (RIM). A Web services registry is realized in RIM version 2.0. The ebXML RegRep Technical Committee expanded its RIM to categorize Web services as RegRep-compatible metadata. The ebXML RegRep treats Web services as a set of metadata that B2B community members want to share with their trading partners. It is designed to support a wide variety of objects and metadata, while UDDI focuses exclusively on Web services. Table 1 summarizes the primary differences between UDDI and ebXML RegReps.

Figure 2 expands on the comparison of UDDI and ebXML by illustrating the differences in structures and metadata (note that the figure is based on ebXML Registry Information Model v2.0).

Web services is one of many possible core technologies needed to realize the goal of collaborative commerce (sometimes referred to as *c-commerce*). Implementing collaborative commerce requires far more than a simple SOAP/UDDI-based infrastructure.

Implementing ebXML RegRep

To implement ebXML RegRep, you must first implement the following ebXML specifications:

- ***For overall architecture:*** ebXML Technical Architecture Specification
- ***For registry:*** ebXML Registry Information Model and ebXML Registry Service Specification
- ***For registry client:*** ebXML Messaging Service Specification
- ***For content:*** ebXML CPP/CPA Specification and ebXML Business Process Specification Schema
- ***For security:*** W3C XML Signature

An illustration of KTNET's ebXML RegRep implementation appears in Figure 3.

Following are some of the new features and improvements in v2.0 of the ebXML RegRep specification:

1. Includes metadata information model of Web services ("Service," "ServiceBinding," "SpecificationLink")
2. Has intramural association and extramural association, depending on the owner of source and target registry objects
3. Separates metadata information model of classification into "ClassificationNode" for child code and "ClassificationScheme" for root node
4. Has "ExternalIdentifier" that must have an attribute, "identification Scheme", that refers a "ClassificationScheme"
5. Has changed the name of "Package" to "RegistryPackage"
6. Has flatter hierarchy of Registry Information Model than v1.0
7. Has SOAP binding about registry service including ebXML messaging service binding
8. No longer provides "Browse and Drill Down Query"
9. Provides only W3C Schema version for registry schema (not XML DTD)
10. Provides two key functional interfaces, "QueryManager" and "LifeCycleManager"
11. Clarifies description of synchronous and asynchronous responses
12. Adds additional status of "LifeCycleManager" (submit, approve, update, deprecate, remove)
13. Clarifies registry communication bootstrapping using URL specified in WSDL for registry
14. Clarifies the specific error handling for LifeCycleManagement protocol

Following is a list of what's missing in ebXML RegRep v2.0:

1. Doesn't provide backward compatibility with v1.0
2. Doesn't provide content-based query
3. Doesn't mention the specific mechanism for cooperative registry (federated registries)
4. Doesn't provide specific method to store core components

The ebXML Registry Technical Committee is currently working on version 3.0 of the specification, following the release of the 2.0 versions of the RIM and RSS (ebXML Registry Service Specification) specifications. Version 3.0 is expected to resolve most (if not all) of the issues in the 2.0 specification.

ebXML RegRep Adoption

The Pan-Asian Alliance was formed in 1999 by five e-commerce service providers:

- ***Infoshare Information Technology Development Co.:*** www.info-share.com.cn (China)
- ***Korea Trade Network (KTNET):*** www.ktnet.com
- ***CrimsonLogic:*** www.crimsonlogic.com.sg (Singapore)
- ***Trade-Van Information Services Co.:*** www.tradevan.com.tw (Taiwan)
- ***Tradelink Electronic Commerce Ltd:*** www.tradelink.com.hk (Hong Kong).

This year TEDI Club (www.tediclub.com/ – Japan) joined PAA as a founding member.

Dagang Net (www.dagangnet.com/ – Malaysia) is an ordinary member.

Businesses from the participating countries can look forward to seamless cross-border trading, with the ready acceptance of cross-border trade approvals and certification policy; the secure and reliable paperless trade and transactions; and, most of all, the efficiency and convenience of working with the alliance as a single point of contact. Combined membership of the parties now exceeds 120,000 organizations, representing almost all active trading enterprises in the Asian market.

The PAA Working Group had several options for building the PAA technical architecture. Through several face-to-face working group meetings and teleconferences, the PAA WG decided to base the technical architecture on ebXML, which was admired for its interoperability, flexibility, extensibility, and reliability. The PAA WG firmly believes that ebXML will show its true merit and prove to be tremendously beneficial in due time. Figure 4 illustrates the PAA technical architecture.

During the first phase, PAA adopted three components of ebXML: messaging service (TRP), CPP/CPA, and RegRep. Security was handled by the adoption of W3C XML signature and Secure Transport (HTTPS). Figure 5 illustrates the capabilities of PAA's pilot project.

PAA's service providers are conducting test interconnections based on the ebXML messaging service 1.0 and CPP/CPA 1.0 specifications.

What's the role of the ebXML RegRep in PAA? First of all, it manages each service provider's CPP/CPA and each trading partner's CPP. If a trading partner is unable to support ebXML, the service provider creates and submits the necessary information on behalf of its customer's CPP.

PAA has developed its own business document standard using DTDs. The alliance publishes its DTDs, sample XML documents, and guidelines to the ebXML RegRep. Other materials for PAA members can also be shared and managed via the ebXML RegRep.

Conclusion

During the pilot project based on ebXML, I realized the difficulty in designing and implementing a single global standard for e-business. It takes time to guarantee conformance to the ebXML specifications and verify interoperability across the trading partners. The workload increases in a geometric progression, depending on the number of partners in the trading community. Moreover, the ebXML RegRep specification is very broad and has a number of optional features. This causes many difficulties when the goal is to reach a consensus among all partners.

While I participated in the design and development of the ebXML RegRep specifications, the level of effort needed to implement it in the "real world" was quite significant.

One more thing to keep in mind: ebXML can't change the world. Despite its popularity, companies will continue to support their legacy standards (such as EDI, XML/EDI, industry-specific standards). Integrating new technical initiatives with legacy standards/environments can pose a significant challenge. Nevertheless, for the future you're going to have to adopt ebXML, step by step as ebXML evolves.

When will the ebXML dream of a single global market for small and medium enterprises become a reality? I daresay it depends on how you adopt ebXML, coupled with a long-term strategy appropriate to your organization.

Resources

ebXML registries

- *Korea Trade Network:* www.GXMLHub.com
- *DISA (Data Interchange Standards Association):* www.disa.org/drive/
- *ebXML Registry Open Source Site:* http://ebxmlrr.sourceforge.net/
- *Hong Kong University:* www.cecid.hku.hk/Release/PR09APR2002.html

CMKIM@KTNET.COM

AUTHOR BIO

Chaemee Kim is a solution manager for KTNET, an electronic trade service provider. Currently Chaemee is building a collaborative trade platform based on ebXML infrastructure for next-generation services. She also leads the Pan-Asian Alliance technical working group project to define and implement global transaction flows within the Asian-Pacific region, and has contributed to several ebXML components.

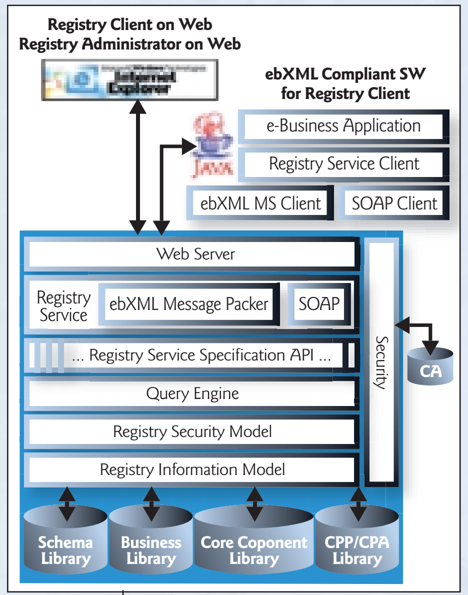


FIGURE 3 | Architectural overview of KTNET's ebXML RegRep

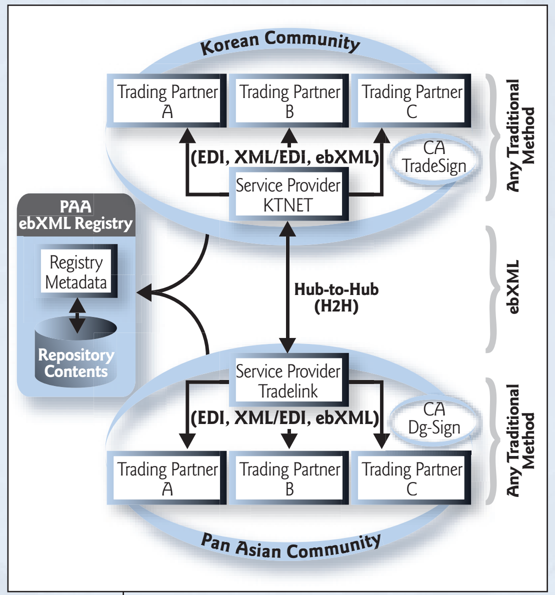


FIGURE 4 | Pan-Asian Alliance technical architecture (based on ebXML)

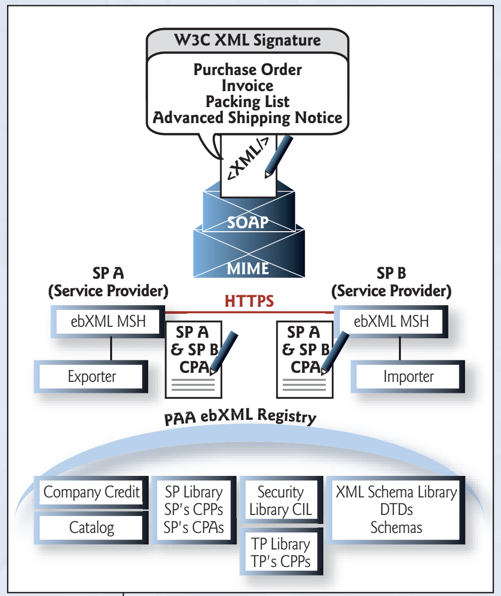


FIGURE 5 | Overview of PAA's ebXML pilot project

WRITTEN BY **JP MORGENTHAL**

An XML Framework for Registry Development

Is there a global framework in the offing?

Over the past few years there has been an emergence of registries due to the lack of organization of the World Wide Web. These registries provide a mechanism to centrally organize information in a way that makes it easier for both people and machines to locate the required sources of information.

However, there is also an alarming new trend taking place now, which is the proliferation of registries, each with a different set of data structures, access mechanisms, and expected functionality. At present there are many parallel registry standards, such as UDDI (Universal Description, Discovery, and Integration), ebXML, .NET Passport, Domain Names Services (DNS), and so on. While these standards aren't necessarily exclusive of one another, each has a different function and audience it intends to satisfy.

I propose that these standards, while seemingly useful relative to their tasks, can all be subordinated to a properly designed Semantic Web initiative. The outcome of such a venture would produce a single methodology for the storage and retrieval of any type of registry information as well as facilitate reusability across registry types. Additionally, highly focused programming interfaces wouldn't be necessary as the single, abstract registry model would support the needs of creating and querying specialized subsets of a larger registry model.

Author Bio

JP Morgenthal is an independent consultant based in northern Virginia. The author of two well-regarded books and numerous articles on XML, EAI, B2B, ERP, XRM, and SCM, JP has served as consultant to Sabre Group, Lockheed-Martin, Citibank, ADP, and Kelloggs, among others.

What Is a Registry?

One of the key attributes of a registry is metadata, which also happens to be one of the key factors of XML that has endeared it to the IT community. A registry is a centralized store that contains individual entries, each representing information (metadata) pertaining to a particular category.

UDDI is an example of a registry for Web services. The registry contains all the information necessary to contact the provider of the Web service and how to bind to the service from an application. Providing this consistent set of information across all entries allows the registry to then provide query facilities to make the registry more useful. However, because UDDI is a focused registry, you won't find information on how to learn Web services or Web service standards.

The UDDI registry also defines specific XML structures that must be understood by any user of the UDDI registry, such as the businessEntity, businessService, and tModel. Each of these structures is defined by the UDDI organization and is proprietary to that registry. That is, they can't be used effectively with other registries, such as ebXML.

The Semantic Web Initiative

The Semantic Web, a W3C Activity (www.w3.org/2001/sw) led by Tim Berners-Lee, James Hendler, and Ora Lassila, is defined by the following quotation: "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

One of the key enabling technologies for the Semantic Web is the Resource Description Framework (RDF), which is another initiative within the Semantic Web Activity within the W3C. RDF has an associated XML syntax for expressing a directed graph of nodes connected by arcs denoted by Universal Resource Identifiers (URI). Listing 1 is an example of an RDF taken from the W3C RDF/XML Syntax Specification Working Draft document that describes the relationship between a Web page and its publisher.

As you can see from the listing, the RDF document creates an association between the editor (Dave Beckett) of the document located at www.w3.org/TR/rdf-syntax-grammar and his homepage (http://purl.org/net/dajobe/). In this way the RDF can create flexible associations between disparate pieces of information.

How the Semantic Web Can Influence Registry Development

The RDF in Listing 2 could also represent the same information set found in the UDDI registry.

In the previous example, what I hoped to prove is that we already have an effective framework for the categorization and association of information identified with the Web infrastructure in RDF. With this framework in place, what we need is a consistent programming interface to allow this information to be managed, replicated, and updated in a consistent fashion. Creation of individual and distinct registries, each with their own structures, query facilities, and programming interfaces, is unnecessary and will lead to downstream hurdles for interoperability, reusability, and heightened restrictions on usage.

Additionally, Query, which seems to be one of the more predominant and interesting tasks being levied against these registries, can be far more useful and less proprietary when an abstract registry representation such as RDF is used. That is, only one search engine would be needed to query the arcs and nodes of an RDF document, and an RDF document could represent, in an equivalent manner, a UDDI business service and an ebXML Collaborative Protocol Agreement.

Conclusion

The resulting work of the Semantic Web Activity can be used to create a global registry framework in which information can be identified by multiple associations without creating barriers to entry, and to provide a single methodology for search and retrieval of information within this network of information.

Likewise, when coupled with standards for creation of ontologies (www.w3.org/2001/sw/WebOnt), the Semantic Web offers the ability to create multiple views of the same set of relationships in varied forms so as to best fit the needs of the user community without the need for specialized registries.

JPM@SYS-CON.COM

LISTING 1

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:ex="http://example.org/stuff/1.0/">
  <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
    <dc:title>RDF/XML Syntax Specification (Revised)</dc:title>
    <ex:editor rdf:parseType="Resource">
      <ex:fullName>Dave Beckett</ex:fullName>
      <ex:homePage rdf:resource="http://purl.org/net/dajobe/" />
    </ex:editor>
  </rdf:Description>
</rdf:RDF>
```

LISTING 2

```
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
xmlns:uddi=" http://www.uddi.org/schema/uddi_v2.xsd">
<rdf:Description
rdf:about="http://www-3.ibm.com/services/uddi/findservice?action=
 details&servicekey= 892A3470-3AAF-11D5-80DC-002035229C64">
       <uddi:name>UDDI Business Registry inquiry</uddi:name>
  <uddi:key>892A3470-3AAF-11D5-80DC-002035229C64</uddi:key>
  Query UDDI Business Registry information
  <uddi:accesspoint
   URL=http://www3.ibm.com/services/uddi/inquiryapi>
   UDDI Business Registry inquiry (SOAP)</uddi:accesspoint>
   </rdf:Description>
</rdf:RDF>
```

Download the Code
www.sys-con.com/xml

WRITTEN BY **ERICH IZDEPSKI**

Ultimate Device Support with CC/PP

What the future holds

The explosive growth of Internet-enabled gadgets has caused a major problem: the current Web infrastructure is simply not ready for them.

The Web has (some) beautiful HTML pages that are at best ugly on microbrowsers…when they can be viewed at all. Most Web servers don't recognize these gadgets or what they can do; they don't know how to share rich, graphic content with them. Those that work rely on the gadget (or worse, its user) to memorize new Web addresses that point to specially built pages intended for viewing on that particular device. Your elaborate Web application doesn't work on the thousands of new gadgets your customers want to use to interact with you. New pages with a sliver of the original content and functionality you just finished building are required. It's actually worse than that: you need new pages for each family of devices – the phones, the pagers, the PDAs, and whatever comes next.

When configured properly, Web servers really do know how to recognize gadgets and send them appropriate data, but the gadgets often mislead Web servers about who and what they are, causing problems. Many use nonstandard (and poorly documented) data. Only very specialized, and hence limited, applications can work with these constraints. New applications need to support all manner of different client devices, from WAP browsers and RIM pagers to PDAs running Palm OS or WinCE, all with varying form factors and computational capabilities.

People have been working on this problem for about seven years. The W3C is the leader in this area. Their work products come under the name "Composite Capabilities/Preference Profiles," or CC/PP.

Out of Chaos…

So how can CC/PP help bring order to this seeming chaos? Through a standard definition called the Device Capability Profile. This profile describes the hardware, software, and even network characteristics of a device. In the simplest case the device sends its profile to a Web server, which reads the profile and uses it to choose the correct content to return (see Figure 1).

To get the most from CC/PP, you can use profiles to allow a single Web page to handle a whole family of gadgets by providing minor customizations of the page dynamically. This isn't the same as providing separate pages for every device (which is bad), nor is it the same as having a single page for all devices (which is a pipe dream).

Fewer pages, logically organized by family, are desirable due to the reduced cost to build and maintain the Web site. Profiles must be written in a language the Web server can both read and understand. The language must have a grammar that a computer can understand and a limited, well-defined vocabulary. CC/PP uses XML as the grammar for representing profiles. To be precise, profile data is described using the Resource Description Format (RDF) and encoded in XML.

The profile also needs a specific vocabulary. This is just a list of terms and what they mean in the limited context of gadget capabilities. Terms are simple, like *color capable* or *screen size*. To organize the terms and make the vocabulary easy for humans to understand, a new concept is useful – that of a profile component. A few sample components are Hardware Platform, Software Platform, Browser, Network Characteristics, and Operating System. A CC/PP vocabulary defines the recognized components, their attributes, and their type information. Listing 1 provides a sample profile containing three components and many attributes (namespace identifiers ignored for clarity).

Now that we can read, write, and understand profiles, we must ensure that it stays that way. Your gadget and my Web server must use the same CC/PP vocabulary in order to communicate effectively. Since supporting a vocabulary requires custom programming, having too many is as bad as having none at all due to the incompatibilities resulting from trying to support too many things.

Figure 2 provides a more detailed example that shows the role of CC/PP in a Web page request.

CC/PP provides the simplest way to get quality information about the capabilities of a client device. A major point of using CC/PP is that the server doesn't need any advance information about gadgets or any other Web browser. You design your pages against the CC/PP vocabulary. Using CC/PP to drive page customization improves the user experience.

Now for the Bad News

Up to this point I've been talking about Web servers knowing CC/PP vocabularies and gadgets sending profiles. This is the future of CC/PP. In reality, profile information is not currently being sent as part of the request, and standard Web servers don't know anything about how to retrieve a profile; even if they could, they wouldn't know what to do with the information.

There is also the problem of matching a gadget to the correct profile. You must use the existing HTTP header data sent by all gadgets to find a suitable profile. Unfortunately, standards for header data are perhaps too liberal, and just about anything and everything shows up in HTTP headers, complicating the task. This doesn't mean that CC/PP can't be used right now – it's just not readily available, and has its challenges.

In order to realize the potential of CC/PP, manufacturers must step up and create Internet-based CC/PP repositories describing their hardware and software. Unfortunately, this will be challenging for manufacturers, since the current information they provide is inadequate, inaccurate, and hard to find…when possible at all. So where will CC/PP information come from?

- ***Mobile phone gateways:*** The gateway has to be able to recognize the phone and find the correct profile for it. The profile may be sent within an HTTP header, or just its address may be included.
- ***New gadgets:*** This requires manufacturers to produce CC/PP-compliant devices or upgrade the software in existing devices.
- ***Web services:*** Once you identify the device you're talking to, you could ask a Web service to return its profile.
- ***You:*** Realistically, you are the only reliable source of CC/PP information today. With some effort you can create your own CC/PP repository. Once you have it and can match gadgets to their profiles, you can start to realize the benefits of CC/PP. And you'll be future-proofing your systems for the day when CC/PP information is ubiquitous.

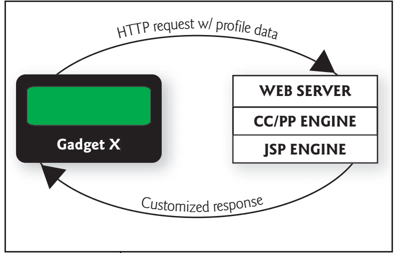


FIGURE 1 | Example Device Capability Profile

Call for Special Software

Let's assume we have a source of CC/PP and can find the right profile for a given gadget. Now what? Handling CC/PP is a task for special software that runs in an application server or a servlet engine. Profiles must be retrieved and turned into server-side objects. These objects must be made available for use in Web pages. JavaServer Pages (including tag libraries) are an ideal choice for harnessing the power of CC/PP to create pages that customize themselves based on a CC/PP profile. Building on the simple profile shown before, Listing 2 provides a very basic example of customization using Cysive's device detection software.

Tag libraries make building a page like this even simpler since common requirements are built into tags, allowing reuse. An added benefit of using tag libraries to harness device information is that a nonprogrammer can use them to build complex pages that will work on a variety of platforms.

With the emergence of CC/PP, some tasks performed by Web applications will be eliminated. The guesswork involved in identifying device capabilities will be no more. Ugly client-side JavaScript used to identify browsers disappears. With accurate profiles, your Web applications will simply know what does and does not work for a given client.

CC/PP has great potential. With it, we will be able to:

- Make Web applications work better.
- Perform advanced device customization.
- Handle new devices automatically, rather than having to update existing application code.
- Enable consumer-oriented Web services to work on any Internet-enabled device.

This is all possible with CC/PP, but more people in the hardware and software communities need to know about it in order for this potential to be realized. Invest in your company's future, and visit the CC/PP Working Group site.

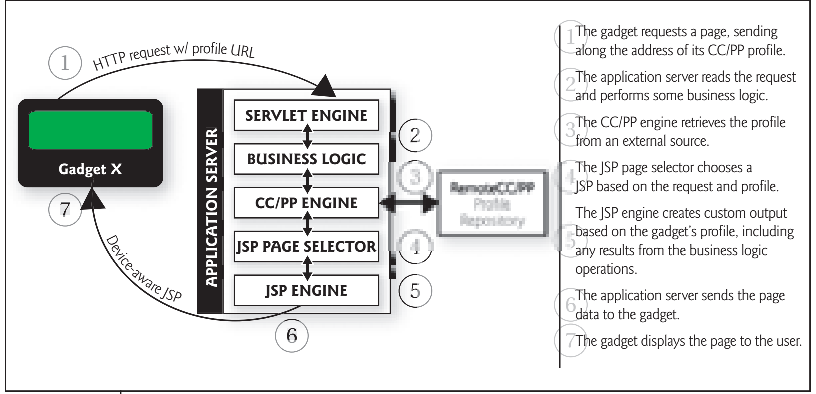

1
HTTP request w/ profile URL
Gadget X
7
Device-aware JSP
APPLICATION SERVER
SERVLET ENGINE
BUSINESS LOGIC
CC/PP ENGINE
JSP PAGE SELECTOR
JSP ENGINE
2
3
4
5
6
RemoteCC/PP Profile Repository
1 The gadget requests a page, sending along the address of its CC/PP profile.
2 The application server reads the request and performs some business logic.
3 The CC/PP engine retrieves the profile from an external source.
4 The JSP page selector chooses a JSP based on the request and profile.
5 The JSP engine creates custom output based on the gadget's profile, including any results from the business logic operations.
6 The application server sends the page data to the gadget.
7 The gadget displays the page to the user.

FIGURE 2 | Role of CC/PP in a Web page request

Reference

- *W3C CC/PP Working Group Web page:* www.w3.org/Mobile/CCPP

EIZDEPSKI@CYSIVE.COM

AUTHOR BIO
Erich Izdepski is a senior developer at Cysive, Inc., based in Reston, VA. Since joining Cysive in 2000, Erich has worked on several projects involving the integration of J2EE and legacy applications. He is currently involved in the development of Cymbio, Cysive's J2EE-based interaction server product.

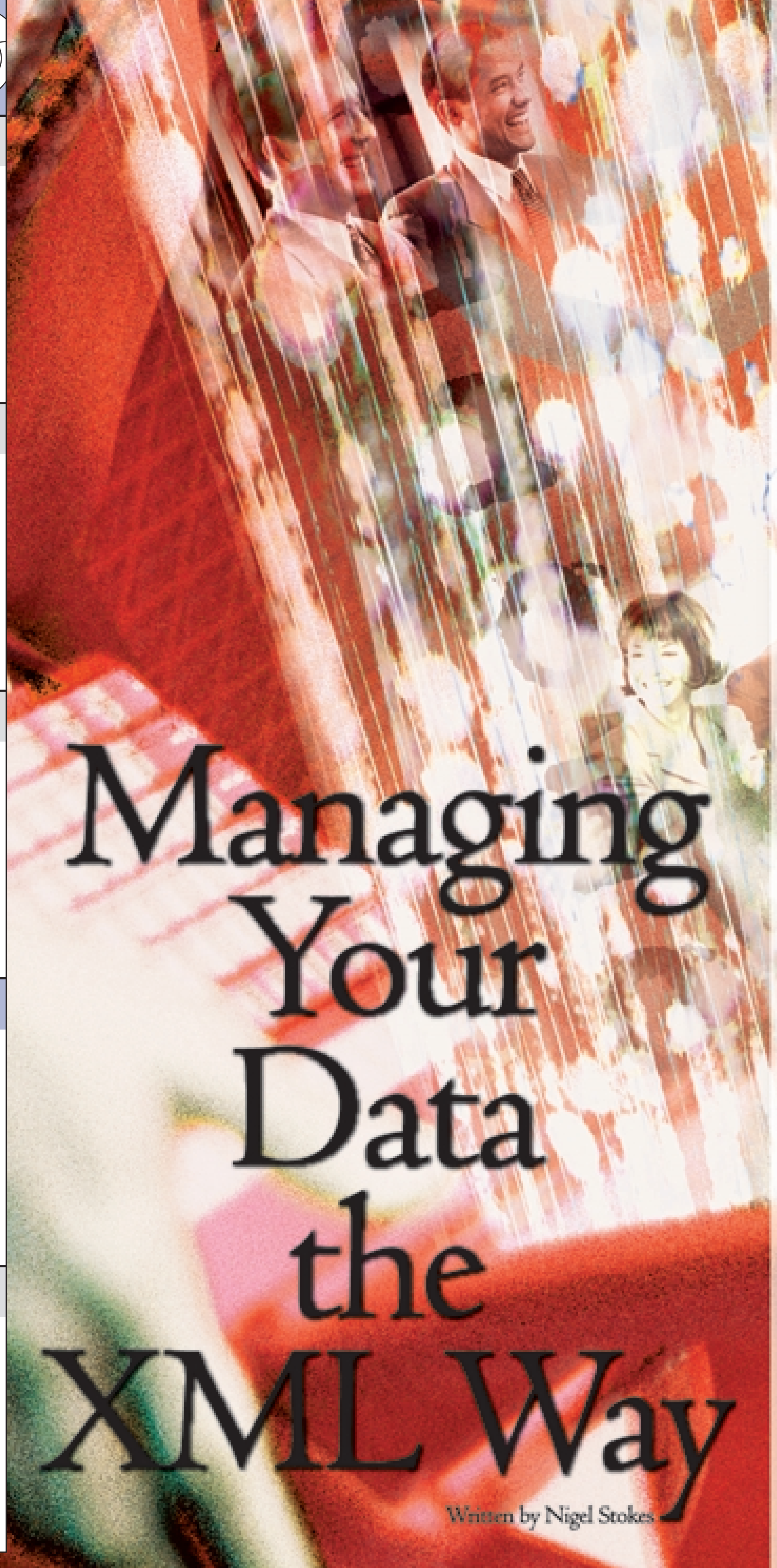
LISTING 1

```
<RDF>
  <Description>
    <component>
      <Description ID="HardwarePlatform">
  <IsColorCapable>Yes</IsColorCapable>
  <ImageCapable>yes</ImageCapable>
  <ScreenSize>160x160</ScreenSize>
      </Description>
    </component>

    <component>
      <Description ID="SoftwarePlatform">
          <OSName>PalmOS</OSName>
      </Description>
    </component>

    <component>
       <Description ID="BrowserUA">
  <HtmlVersion>3.2</HtmlVersion>
          <TablesCapable>yes</TablesCapable>
       </Description>
     </component>
  </Description>
</RDF>
```

LISTING 2

```
--
IF Device is color capable
THEN show color graphic;
ELSE
show black and white graphic;

IF Device screen width < 640 pixels
THEN show short content for PDA;
ELSE
show long content for PC;
--
```

Download the Code
www.sys-con.com/xml



Managing Your Data the XML Way

Written by Nigel Stokes

The holy grail of e-business and Internet technologies is the ability to enable cost-effective business-to-business transactions with trading partners, eliminating the inefficiencies of paper trails, duplicate data processing, and rekeying of information. Companies are increasingly calling on IT departments to develop a network of relationships encompassing suppliers, business partners, and internal activities that is accessible, integrated, scalable, and resilient.

To stay competitive, companies must be able to transform and exchange business data freely and dynamically between their own systems and those of their business partners via the Internet. This involves integration of many corporate databases, applications, Web sites, and business partners.

Web services promise to provide a way to more easily and cost-effectively exchange, transform, and access corporate data on many different incompatible systems, both within and beyond the walls of the organization. The Web services concept uses the Internet and an open set of standards based on XML as the common data interchange format.

But before B2B or Web services can be considered, an organization may have to deal with significant internal challenges to ensure the interoperability of various systems and applications as well as the integration of various sources of critical business data. Corporate entities generate vast amounts of data every day – transactions, inventory, product information, order status, sales and customer data, invoices, payments, business contacts, reports, statistical data, Web access logs, and so on. In most cases data is stored and managed by various database systems. As businesses grow over time or rapidly evolve through mergers and acquisitions, they acquire many heterogeneous systems that may not easily communicate with one another.

As companies try to steer a course through the hype and develop strategies for implementing B2B and Web services, they will face the challenge of integrating these existing relational databases, EDI infrastructures, and text formats with XML. Real-time XML-driven middleware solutions will play an increasingly important role in helping companies integrate their existing data assets with XML to enable effective B2B relationships.

The role of data integration middleware in enabling effective B2B

B2B Hurdles

For B2B to realize its potential, transactions between business partners must be conducted electronically in an interactive manner to eliminate tedious manual processes such as rekeying information. The challenge is that no two companies have the same database structure or format their data in the same way. The ways in which we store and process data are often so diverse that either:

1. The business partner would have to deliver the data in many different formats *OR*
2. The subscribers or recipients of the data would have to transform the data in order to use it effectively *OR*
3. They'd have to adopt the same systems as the data publisher and a universal standard for data exchange.

The conventional method of electronically exchanging data is to write custom programs to generate and read data in a format agreed to by two parties or a group of participants. When one partner needs to change their database or data structure, either the company will have to rewrite programs to meet the agreed-upon standards, or all other partners who exchange data with this company will be forced to make changes to their own programs accordingly. Adapting to ongoing change results in very high maintenance costs. Companies also have to maintain skilled development personnel in-house or outsource development.

Some large enterprises have implemented sophisticated EDI solutions through a specialized value-added network (VAN). The traditional EDI infrastructure is not only extremely inflexible and expensive to implement, but can also result in high maintenance costs when adapting to ongoing business changes. For small to medium-sized businesses, EDI is often beyond reach. Another major problem with traditional EDI is that it requires a unique solution for each pair of trading partners, resulting in a high transaction cost.

XML presents a more efficient solution for data exchange through value-added transformation and by leveraging the Internet or internal networks connected to enterprise databases in a secure environment. Rather than force data subscribers or receivers to adapt or transform the data format the sender chooses, the data publisher or sender can deliver data in XML that the subscriber or business partner can easily retrieve, view, and use for whatever purpose.

Strengths of XML

Using XML, virtually any data items can be identified, allowing Web pages to function like database records and paving the way for a truly transactional Internet. XML doesn't just relay the data that needs to be transported, it contains information, or metadata, about the data that it's sending. This means that the data is absolutely self-contained within one package with no need to agree on columns, rows, or other data formats or syntax. By providing a common method for identifying data, XML effectively supports B2B transactions. Some of the key features of XML are summarized below.

Extensibility

Using XML, information publishers can define new tags and attribute names at will. Rather than being constrained to defining a particular set of data, XML is able to work with DTDs and schemas to define any number of documents that form a language of their own. Hundreds of thousands of specific document vocabularies have already been created using XML to meet the specific needs of various industries.

Structure

XML was designed for maximum expressive power and is applicable to almost any data or document structure no matter how complex. With XML, document structures can be nested to virtually any level of complexity, allowing the representation of database and object-oriented hierarchies.

Validation

XML documents come with built-in error and validity checking. They may contain a description of its grammar for use by applications that need to perform validation of the information contained in the document before populating a database or executing a transaction. For example, the DTD or schema can be used to verify that all elements are correctly specified, in the right order, and that values fall within acceptable predetermined ranges.

Loosely coupled architecture

XML merely represents information and metadata about the information without indicating any specific way in which the data should be processed or putting any limitations on how the information may be handled or displayed. Because XML separates data from processing, it offers huge benefits for companies seeking a multiplatform, multidatabase solution for B2B.

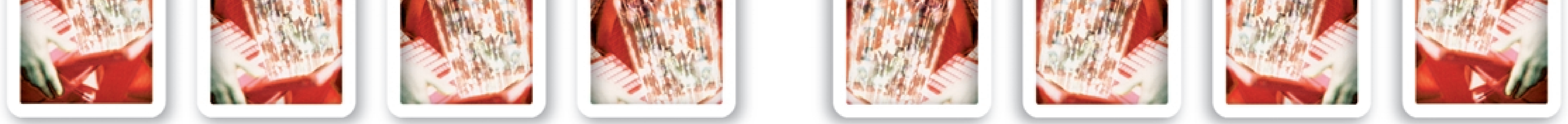
XML documents can be freely exchanged across multiple platforms, databases, and applications so long as the subscriber data stores and applications are XML-aware. Any system, mobile device, or application that speaks XML can access and manipulate the data contained in the XML document at will. This makes XML ideal for enabling B2B where companies often face the challenges of mixed-system environments. Systems built on a loosely coupled XML integration framework are future-proof because changes to the computing environment won't affect XML data exchange.

The unique capabilities of XML, including vendor and system independence, make it an ideal enabling technology for applications that require data to be communicated between two or more heterogeneous databases within the same company or beyond the walls of the organization to include business partners or suppliers.

Challenges of XML Adoption

Since its introduction, few technologies have been hyped as much as XML. But while XML is seen by many as a key B2B enabler, it's really only the first step. XML tags provide a simple data format, but the intelligent defining of these tags and common adherence to their standard usage will determine the real long-term business value of XML.

Already there are several different XML schemas being promoted by different industry coalitions or bodies, including W3C, OASIS, and BizTalk, as well as Web sites that provide repositories (Biztalk.org and XML.org) for publishing and reviewing XML schemas. Hundreds of XML standards have been created since the introduction of XML, which raises the question of whether or not a



standard for XML actually exists. XML standards are currently in flux. The potential for the creation of many different "proprietary" XML documents threatens to undermine the successful adoption of a true XML standard and hinder the widespread deployment of XML solutions across the enterprise.

Role of Data Integration Middleware

The successful adoption of XML for data transformation, exchange, and integration depends on the support of leading integration middleware vendors who can provide out-of-the-box solutions for integrating existing or legacy systems that may not be XML-aware, and ensuring that users can easily and cost-effectively transform data between other data formats and XML as well as between different competing XML standards should this be necessary.

The need for a certain amount of transformation between document formats is inevitable. XML-based integration middleware is necessary to enable users to freely and seamlessly flow and transform data between text, database, and XML. This technology ensures that XML-aware and non-XML-aware systems can coexist, enabling companies to realize the full potential for XML data exchange and integration. XML-powered integration middleware has a role to play in brokering B2B data transactions between legacy applications and application server technologies to enable companies to tightly integrate all their systems into a cohesive infrastructure without requiring custom development or changes at the application layer.

Enterprise data is typically structured as repeating sets of hierarchical entities such as those stored in a relational database. Traditional transformation engines often scale up very well but lack flexibility and cannot deal with hierarchical data. Emerging XML middleware technologies provide an excellent vehicle for developing a new generation of scalable and flexible transformation engines.

XML-powered data integration middleware plays a key role in enabling effective B2B communications by bridging the heterogeneous databases, EDI infrastructures, and text formats deployed by your organization and by your business partners (see Figure 1). XML middleware adds value to B2B implementations by allowing you to leverage your existing mix of systems, including technologies that may not be XML-aware.

For XML-based transformation engines, both DOM and SAX models have substantial limitations in handling enterprise data, scalability, and flexibility. The most advanced middleware technologies take the advantages of both SAX and DOM models and intelligently apply them globally or locally, wherever they're required, to best achieve the desired results. This processing model not only provides the best performance and scalability but also maximizes flexibility.

Choosing a Middleware Solution: Selection Criteria

The middleware market is growing increasingly cluttered with multiple vendors, solutions, and approaches vying for your attention.

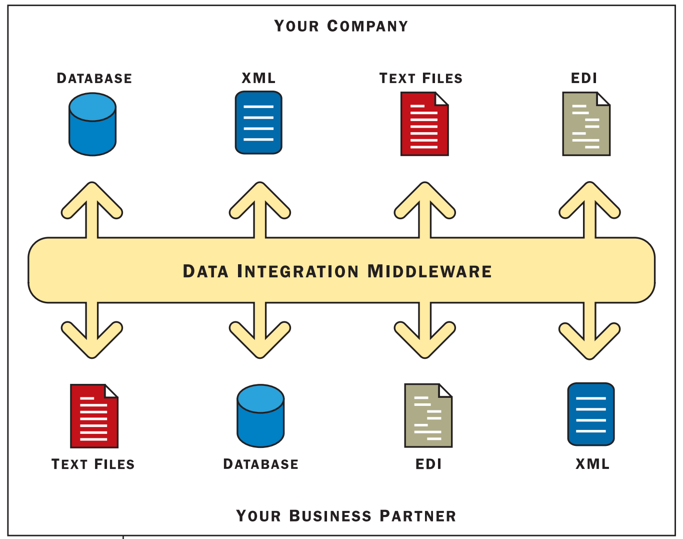


FIGURE 1 | A bridge to effective B2B communications

How can your business see through the clutter and choose an effective middleware solution to enable your B2B initiatives today and in the future?

The following considerations can be used to guide you in your search for an effective middleware solution to connect your internal systems and data sources, integrate your data assets with XML, and pave the way for successful B2B and Web services projects.

Bidirectional, multidatabase data transformation

In a business-to-business environment, data interchange is the norm. You receive XML documents from your business partners or another business unit within your own organization and you want to automatically save the data from those XML documents to your database with your special requirements. The middleware tool should have the capability to transform and flow data bidirectionally between database, XML, EDI, and text formats in any combination to leverage all information across your enterprise. Through XML, data can be transformed from any format to any other format for powerful distributed data applications. XML documents created by the middleware solution should contain not only data but also the relationships of the data. This lets you map the objects between your XML document and database with value-added transformation rules, and update your database accordingly.

Hierarchical data transformation and hierarchy reconstruction

The tool should allow you to transform hierarchical data between database and XML, database and database, XML and XML. The tool should read and map databases to and from XML according to their hierarchical relationships. It should maintain relationships of data by reconstructing the hierarchy and mirroring the relationships in the original database. Look for a middleware solution that can automatically format a flat query result into a hierarchical structure, allowing administrators to easily configure and map data transformations.

Customizable rules and data manipulation

For maximum flexibility the middleware tool should be able to manipulate the data itself as well as the data structure or format to generate unique key values required by the database. Administrators should have the ability to incrementally update databases, add extra fields required by the target, assign fixed or dynamic default values to fields, completely change data structures, and apply formatting rules to data.

Support for XPath expressions and functions

XPath expressions and functions are used to map the data source for cross-level object mapping as well as mixed mapping between element and attribute. The middleware solution should support built-in XPath and complex expression mappings, as well as complementary functions to XPath and XSLT such as date formatting, string conversion, and obtaining system date and time. It also should accommodate database-specific extensions for stored procedure calls, data lookup, and generating key values using the database's stored procedures and sequences as well as Java objects.

Open architecture

Open architecture enables users to plug in Java objects for complex processing requirements, such as data validation and translation with lookup tables. Open API class libraries and documentation enable third-party applications to easily interact with the middleware solution.

Built-in query capabilities

To transform data and generate reports, companies have typically written custom-coded programs. These programs can't be easily shared or reused by different applications or systems. This means that enterprises have to write and maintain a program for each usage scenario and for any change in the computing environment. Custom programs developed in-house lack the ability to keep pace with the changing needs of enterprise systems. Developers have to constantly rewrite the report generator program to adapt to changes. The result is a very high and ongoing maintenance cost. Another issue enterprises often encounter in the generation of reports is that complex queries can cause considerable overhead on server resources, significantly slowing down key business processes and, in the worst-case scenario, resulting in system downtime. To overcome these potential problems, some companies summarize the data at off-peak times after normal business hours. While this batch process resolves the resource problem for production servers, it creates a new problem: the data used to generate reports is stale and out of date. The middleware solution should have built-in support for running effective queries, report generation, and real-time analytics.

Summing Up

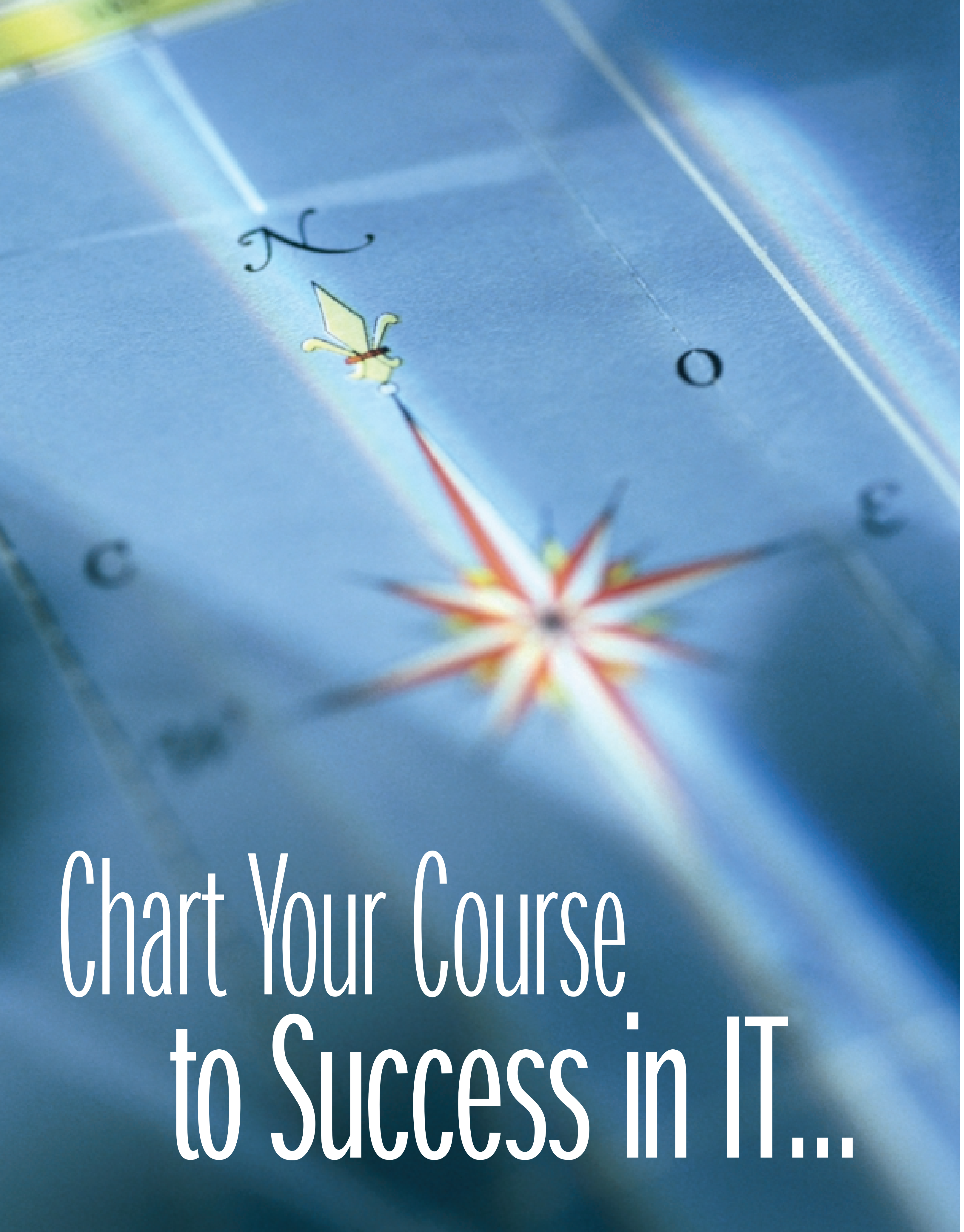
Wise investments in XML, as well as XML-powered integration middleware, can effectively future-proof your business against many of the pitfalls of changing information systems, best-of-breed applications, and multiplatform computing environments. Moreover, XML promises to provide a powerful tool for building bridges between your organization and your business partners and suppliers that will help you achieve significant business returns. The next generation of XML solutions will enable true real-time data exchange between diverse systems, both in-house and beyond, to encompass business partners and suppliers.

An effective XML-based B2B strategy can help your business realize new operational efficiencies while enhancing service levels and maximizing partnership and revenue opportunities. New middleware tools that support open standards like XML are well worth evaluating. Thorough testing and proof-of-concept will determine whether these technologies are a good fit with your organization's data infrastructure.

AUTHOR BIO

Nigel Stokes is CEO of DataMirror, a leading provider of enterprise application integration and resiliency software solutions. Over 1,600 customers worldwide use DataMirror software.

NSTOKES@DATAMIRROR.COM



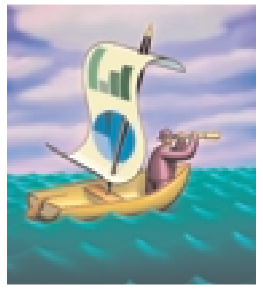
...order your copy of

Java Trends: 2003

Available July 1*

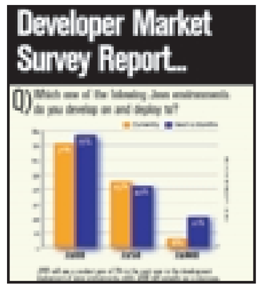
EnginData's research is invaluable to leading IT vendors, integrators, Fortune 500 companies, trade-show organizers, publishers, and e-business organizations worldwide.

Don't go astray. In the vast sea of Internet technology, market conditions change constantly. Will Java remain the hot platform it is today? Will C# rapidly gain ground? What are the world's foremost Java developers aiming their sights toward? Which companies come to mind when planning their next project? How can their thinking direct your efforts?



EnginData Research has studied the IT industry extensively, spotting many key trends and alerting industry leaders along the way. In the first quarter of 2002, they embarked on their most challenging mission ever: the most in-depth study ever done of the Java development marketplace.

After months of analyzing and cross-referencing more than 10,500 comprehensive questionnaires, the results are now available.



Here's just a sample of the critical data points the Java report delivers...

- ✔ Current and projected usage of languages and platforms
- ✔ Multiple rankings of hundreds of software vendors and tools
- ✔ Types of applications being developed
- ✔ Databases being used
- ✔ Purchasing patterns
- ✔ Company size
- ✔ Development and purchasing timelines
- ✔ Perceptions and usage of Web services
- ✔ Preferred Java IDE
- ✔ J2EE, J2ME, J2SE usage comparisons

As an IT specialist, **EnginData Research** focuses exclusively on three leading drivers in IT today – Java, Web services, and wireless development. Coming soon, our Web services survey will deliver such critical knowledge as:

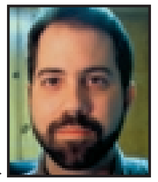
- ✔ Time frame for developing Web services – enabled apps
- ✔ Percentage of apps with Web services today
- ✔ Sourcing and hosting Web services
- ✔ Perceived leaders in Web services tools and solutions

Navigate your company through the challenging IT marketplace with the trusted knowledge and intelligence of **EnginData Research**. Order your copy of the 2002–2003 Java market study by calling Margie Downs at 201-802-3082, or visiting our Web site.

engindata RESEARCH

www.engindata.com

*A limited number of preview copies will be available at our booth (#624) at JDJEdge International Java Developer's Conference & Expo, June 24–27, at the Jacob Javits Convention Center in New York.



WRITTEN BY **ROY HOOBLER**

Searching XML Files with XSLT

How more advanced XSLT techniques can be used

XSLT is generally used to parse and translate XML files, but with some more advanced techniques, it's possible to search for specific attributes (or elements) of any XML document or list of documents.

While developing a search tool, I learned how to replace strings and translate characters (uppercase to lowercase) in XSL. After a little research, it took only a few hours to put together what I needed. The examples in this article show how to implement a simple search mechanism to search a DocBook file and display the search results in HTML format.

When XML first appeared, much of the hype was about how well suited it is for searching documents (and it is – but I haven't seen many implementations). I'd been working on a Web site and wanted to add a search box. Using XML, searching the site should be easy enough. The site itself is in one DocBook XML document. Articles (posts) can be posted to the Web site as well. These are each stored in a separate DocBook Article XML file. Yet another XML file on the server contains a list of these article files, which more or less functions as an index for the directory.

More specifically, the requirements I came up with were:

1. Allow the user to enter a keyword into a text box.
2. Choose to search for any keyword, just author names, or just titles.
3. Scan the Web site content *or* scan the articles.
4. Display the results, showing text around the found keyword and a link to the page or article.
5. Show how many times the keyword was found.
6. Highlight the keywords in the above results.

Design Overview

For those who've seen the examples for Microsoft's Index Server, I was going after the same thing: a search result showing where the keyword was found, with a link to the document (or chapter and section in the case of the DocBook). The final result of searching the Web site is shown in Figure 1, which displays the search box with the options to search for content or titles (searching by author is for the "Article" search).

Since the Web site uses the DocBook Book and the DocBook Article formats, I separated the functionality and made two different search templates. This article focuses mainly on the DocBook format and the showSearch template (see Listing 1). However, both formats have a similar structure after the root element, so the Author, Title, and Paragraph templates (see Listing 2) can be used with either search. On the home page the main search box searches the site's content or just page titles (the DocBook). If a user is viewing a list of articles in a category, there's another search box to search all articles.

Designing the XSLT templates

The first template accepts four parameters, everything we need to start searching: what part of the document to search, a keyword, a path, and a filename. Declared at the top of the first XSL stylesheet, these parameters are available to any template whether listed in the current file or in an included stylesheet (see Listing 3). Next, the showSearch template performs the search using apply-templates and the XPath query:

```
$work/book/descendant::*[contains(tex
t(),$searchString) and
name()=$searchType]
```

Here the variable $work contains our document. The rest of the XPath query will actually find matching text() nodes containing our search keyword. The query will also filter to match only certain nodes we specify (author nodes, title nodes, or paragraph nodes) by using the $searchType variable. The technique for getting the XPath to search the entire document is to use the descendant::* axis.

What is an "axis"? Performing complex XSLT processing usually requires the use of axes. This example is fairly simple: matching nodes under the <book> node. Other axes retrieve the value of the next node, previous node, current node (self), and parent node. So, if you want to return only books or articles with authors (assuming some don't have an author), you can use a template to match authors and use the parent axis to retrieve the book's title. Many of the axes have shorthand counterparts such as ".." for parent node and "." for the current node. I've found, however, that the shorthand doesn't work as well as using the axis by name.

The XPath is the only difference between searching articles and searching books. For article searches the XPath simply needs to be changed to $work/article/descendant::*.

With the above query, we can return results, but XML and XSL are case sensitive, so searching for "cad" won't return any elements containing "CAD". After a little research I came across the XSLT translate function, which is designed to change characters from one format to another. In this case all uppercase letters will be translated to lowercase before the query is executed. To use the translate function, two variables containing characters are created, one lowercase and one uppercase, and placed on top of the stylesheet.

The XPath query and the translate function use these variables, swapping any character from one set (the uppercase) to the other (lowercase) for both the search term and elements (see the declarations in Listing 4). The query is run twice, once to get the count of how many matches were found and once as the select statement of the apply-template element. Doing this type of operation takes more processing power, but I've been surprised by how quickly the query is performed.

The final XPath query in the apply-templates element looks like:

```
<xsl:apply-templates
select="$work/book/descendant::*[con-
tains(translate(normalize-
space(text()),$ucletters,$lclet-
ters),translate($searchString,$uclet-
ters,$lcletters)) and
name()=$searchType]" mode="search"/>
```

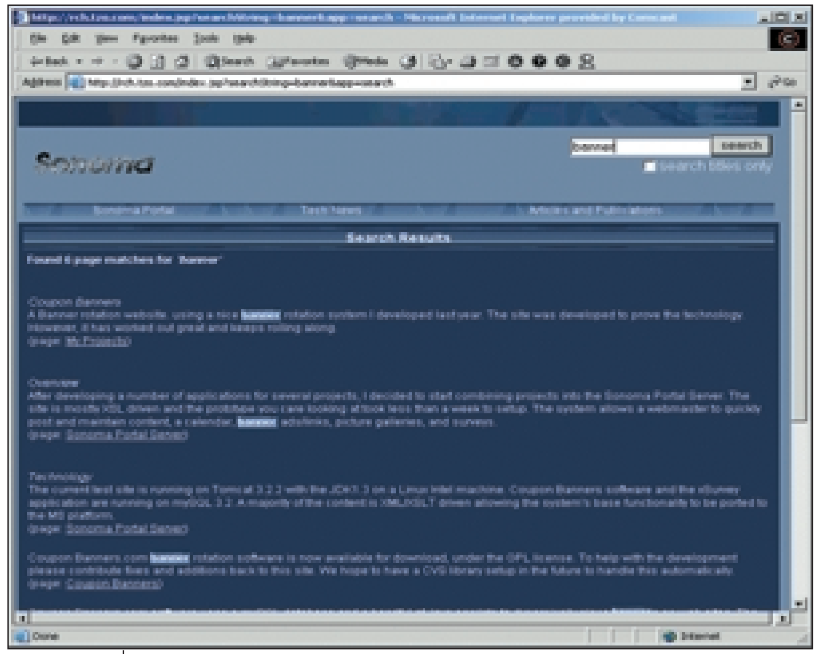
Displaying the search results

Getting the apply-templates to match the correct elements was only half the battle. Along the way, I started to build templates to display what was being returned. The XSL templates in Listing 2 match and display the Text() element of each Author, Title, and Paragraph where a keyword was found. The structure of "DocBook" is probably pretty good for searching since all the content is laid out in a fairly strict format. I've kept things simple for the examples in this article; as more elements are introduced, these templates will become more complex.

Highlighting Keywords

The next step (and the second new XSLT function I had to implement) was to highlight keywords displayed within the search results. The replace-string template (see Listing 5) is similar to a recursive method, continuously parsing the text until all the keywords have been replaced.

I've learned that using recursive techniques is a very common and pow-



Search Results

Found 6 page matches for 'banner'

Coupon Banners
A Banner rotation website, using a nice banner rotation system I developed last year. The site was developed to prove the technology. However, it has worked out great and keeps rolling along.
(page: My Projects)

Overview
After developing a number of applications for several projects, I decided to start combining projects into the Sonoma Portal Server. The site is mostly XSL driven and the prototype you are looking at took less than a week to setup. The system allows a webmaster to quickly post and maintain content, a calendar, banner ads/links, picture galleries, and surveys.
(page: Sonoma Portal Server)

Technology
The current test site is running on Tomcat 3.2.2 with the JDK1.3 on a Linux Intel machine. Coupon Banners software and the eSurvey application are running on mySQL 3.2. A majority of the content is XML/XSLT driven allowing the system's base functionality to be ported to the MS platform.
(page: Sonoma Portal Server)

Coupon Banners.com banner rotation software is now available for download, under the GPL license. To help with the development please contribute fixes and additions back to this site. We hope to have a CVS library setup in the future to handle this automatically.
(page: Coupon Banners)

FIGURE 1 | Search box with options to search for content or titles

Author Bio

Roy Hoobler has been developing custom Web applications since 1996. After completing his MCSD certification, he spent the mid-'90s at a large consulting firm focused on intranet/extranet applications for Fortune 1000 companies. In 1998 Roy joined Net@Work (www.netatwork.com) as director of Internet technologies, specializing in systems architecture, project management, and research into emerging programming methods.

erful tool in XSLT. The whole idea around transformations and using a document tree relies heavily on recursive structures and implementations.

Placed at the beginning of the stylesheet document (under all the other variables), the variable $myReplacedText holds the complete value of the HTML text that will be replaced. To keep things simple, the application can accept only one keyword or exact phrase per search. Since $ myReplacedText is at the beginning of the document, it can be used in any template throughout the XSL stylesheet. After this variable is added, the completed XSL declarations look like the snippet in Listing 4.

In Listing 5 the recursive string-replace template replaces keywords found in the document with the keyword itself, surrounded by an HTML <font> tag.

Now that the XSL templates are done, the fun can begin. In an ASP or JSP file, load the XML document, create an XSLT parser, and set the parameters for $searchString and $searchType. In my final application I used another XML file and XSL template that listed the documents in a directory containing Articles. The first template actually looped through the list of articles, performed the same search, calling a showArticleSearch template, and passed the parameters and document for each article in the list.

Summary

With the translate function, descendant axis, and replace template, building a small search utility gave me some exposure to how more advanced XSLT techniques can be used. I've implemented this type of searching on a couple of Web sites and the performance is pretty good. It isn't a complete solution, but it adds a nice feature. The listings and examples here are taken from a larger solution. If anyone would like a complete set of working files, please send me an e-mail.

RHOOBLER@NETATWORK.COM

LISTING 1 The main search template

```
<xsl:template name="showSearch">
<xsl:variable name="work"  select=
 "document(concat($filesDirectory,'/',$xmlfile))"/>
<xsl:if test="$searchString!=''">
 <span class="normtext"><b>
  Found <xsl:value-of select="count($work/book/descendant::
  *[contains(translate(text(),$ucletters,$lcletters),
  translate($searchString,$ucletters,$lcletters)) and
  name()=$searchType])"/> page matches for <i>'
  <xsl:value-of select="$searchString"/>'</i><br/></b></span>
 <xsl:apply-templates select="$work/book/descendant::*
  [contains(translate(normalize-space(text()),$ucletters,
  $lcletters),translate($searchString,$ucletters,
  $lcletters)) and name()=$searchType]" mode="search"/>
</xsl:if>
</xsl:template>
```

LISTING 2 Templates for displaying the results

```
 <xsl:template match="para" mode="search">
 <xsl:variable name="StringReplaced">
     <xsl:call-template name="replace-string">
       <xsl:with-param name="text" select="text()"/>
     </xsl:call-template>
  </xsl:variable>
  <p  class="normtext">
<xsl:apply-templates select="child::*" mode="search"/>
<xsl:value-of select="$StringReplaced"
 disable-output-escaping="yes"/> <BR/>
<xsl:if test="$topic=''">
  (page: <a href="index.jsp?page={../@id}&amp;ch=
   {../../@id}"><xsl:value-of select="../@label"/></a>)
</xsl:if>
  </p>
 </xsl:template>


 <xsl:template match="TITLE" mode="search">
 <xsl:variable name="StringReplaced">
     <xsl:call-template name="replace-string">
       <xsl:with-param name="text" select="."/>
     </xsl:call-template>
  </xsl:variable>

<i><xsl:value-of select="$StringReplaced"
 disable-output-escaping="yes"/></i> (Book) /
 <xsl:apply-templates select="../AUTHOR"/>
<br/>
 </xsl:template>

 <xsl:template match="AUTHOR" mode="search">
<xsl:value-of select="."/> (Author)
 </xsl:template>
```

LISTING 3 Accepting parameters

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl=
 "http://www.w3.org/1999/XSL/Transform"
<xsl:param name="filesDirectory"></xsl:param>
<xsl:param name="xmlfile">test2.xml</xsl:param>
<xsl:param name="searchString"/>
<xsl:param name="searchType">para</xsl:param>
<!-- load the document, use with param to call
 showSearch template -->
 <xsl:template match="/">
  <xsl:call-template name="showSearch"/>
 </xsl:template>
</xsl:stylesheet>
```

LISTING 4 XSL variable declarations / match the root

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl=
 "http://www.w3.org/1999/XSL/Transform" xmlns=
 "http://www.w3.org/TR/REC-html40" version="1.0">
<xsl:variable name="lcletters">abcdefghijklmnopqrstuvwxyz
 </xsl:variable>
<xsl:variable name="ucletters">ABCDEFGHIJKLMNOPQRSTUVWXYZ
 </xsl:variable>
<xsl:variable name="myNewString"><xsl:value-of select="concat
 ('&lt;font class=foundText &gt;&lt;b&gt;' ,
$searchString ,
 '&lt;/b&gt;&lt;/font&gt;')"/></xsl:variable>
```

LISTING 5 XSL replace-string template

```
<xsl:template name="replace-string">
    <xsl:param name="text"/>
    <xsl:param name="replace"/>
    <xsl:param name="with"/>
    <xsl:choose>
      <xsl:when test="contains($text,$replace)">
        <xsl:value-of select="substring-before
          ($text,$replace)"/>
        <xsl:value-of select="$with"/>
        <xsl:call-template name="replace-string">
          <xsl:with-param name="text"
  select="substring-after($text,$replace)"/>
          <xsl:with-param name="replace" select=
           "$searchString"/>
          <xsl:with-param name="with" select=
           "$myNewString"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$text"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
```

Download the Code
www.sys-con.com/xml

Simplex Knowledge

www.skc.com

FREE TUTORIAL
JUNE 27TH WITH WEB SERVICES EDGE 2002 EAST REGISTRATION. LIMITED OFFER!

TO REGISTER: www.sys-con.com or Call 201 802-3069

web services EDGE world tour 2002

$195 REGISTRATION FOR SYS-CON SUBSCRIBERS
BEST EDUCATIONAL VALUE FOR THE HOTTEST TECHNOLOGY SKILLS!

Take Your Career to the Next Level!

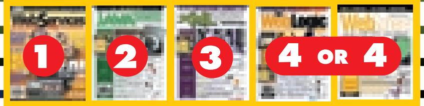
SEATING IS LIMITED. REGISTER NOW FOR THE CITY NEAREST YOU! WWW.SYS-CON.COM

Learn How to Create, Test and Deploy Enterprise-Class Web Services Applications

TAUGHT BY THE INNOVATORS AND THOUGHT LEADERS IN WEB SERVICES

EXPERT PRACTITIONERS TAKING AN APPLIED APPROACH WILL PRESENT TOPICS INCLUDING BASIC TECHNOLOGIES SUCH AS SOAP, WSDL, UDDI AND XML, PLUS MORE ADVANCED ISSUES SUCH AS SECURITY, EXPOSING LEGACY SYSTEMS AND REMOTE REFERENCES.

RECEIVE 4 FREE 1-YEAR SUBSCRIPTIONS
$345 VALUE!



SHARPEN YOUR PROFESSIONAL SKILLS.

KEEP UP WITH THE TECHNOLOGY EVOLUTION!

"Presented an excellent overview of Web services. Great inside knowledge of both the new and old protocols. Great insight into the code piece."
– Rodrigo Frontecilla

"Very articulate on the Web services SOAP topic and well-prepared for many questions. I've learned a lot from this seminar and I appreciate this seminar for my job. Thank you!"
– Kenneth Unpingco, Southern Wine & Spirits of America

"I liked the overview of Web services and the use of specific tools to display ways to distribute Web services. Good for getting up to speed on the concepts."
– B. Ashton, Stopjetlag.com

Echoed over and over by Web Services Edge World Tour Attendees:

"Good balance of theory and demonstration."

"Excellent scope and depth for my background at this time. Use of examples was good."

"It was tailored toward my needs as a novice to SOAP Web services – and they explained everything."

WHO SHOULD ATTEND:

- **Architects**
- **Developers**
- **Programmers**
- **IS/IT Managers**
- **C-Level Executives**
- ***i*-Technology Professionals**

IF YOU MISSED THESE...

BOSTON, MA (Boston Marriott Newton) SOLD OUT!
WASHINGTON, DC (Tysons Corner Marriott) SOLD OUT!
NEW YORK, NY (Doubletree Guest Suites) SOLD OUT!
SAN FRANCISCO, CA (Marriott San Francisco) SOLD OUT! CLASSES ADDED

BE SURE NOT TO MISS THESE...

Each city will be sponsored by a leading Web services company

...COMING TO A CITY NEAR YOU

2002

NEW YORK AT WEBSERVICES EDGE 2002 EAST	JUNE 27
BOSTON	JULY 10
SAN FRANCISCO	AUGUST 6
SEATTLE	AUGUST 27
AUSTIN	SEPTEMBER 10
LOS ANGELES	SEPTEMBER 19
SAN JOSE AT WEBSERVICES EDGE 2002 WEST	OCTOBER 8
CHICAGO	OCTOBER 17
ATLANTA	OCTOBER 29
MINNEAPOLIS	NOVEMBER 7
NEW YORK	NOVEMBER 18
SAN FRANCISCO	DECEMBER 3

2003

CHARLOTTE	JANUARY 7
MIAMI	JANUARY 14
DALLAS	FEBRUARY 4
BALTIMORE	FEBRUARY 20
BOSTON	MARCH 11

REGISTER WITH A COLLEAGUE AND SAVE 15% OFF THE LOWEST REGISTRATION FEE.

TOPICS HAVE INCLUDED:
Developing SOAP Web Services
Architecting J2EE Web Services

On May 13th, the San Francisco tutorial drew a record 601 registrations.

SYS-CON *i*-TECHNOLOGY EDUCATION

REGISTRATION FOR EACH CITY CLOSES THREE BUSINESS DAYS BEFORE EACH TUTORIAL DATE. DON'T DELAY. SEATING IS LIMITED.
NON-SUBSCRIBERS: REGISTER FOR $245 AND RECEIVE THREE FREE ONE-YEAR SUBSCRIPTIONS TO *WEB SERVICES JOURNAL, JAVA DEVELOPER'S JOURNAL*, AND *XML-JOURNAL*, PLUS YOUR CHOICE OF *BEA WEBLOGIC DEVELOPER'S JOURNAL* OR *WEBSPHERE DEVELOPER'S JOURNAL*, A $345 VALUE!

TO REGISTER: www.sys-con.com or Call 201 802-3069

WRITTEN BY **STEVE HOENISCH**

Switching Document Views

Applying XSLT stylesheets dynamically with DOM

Complex technical documentation presented on the Internet calls for user interfaces or navigational options that empower readers to quickly gain access to the information that suits their needs. If your readers are viewing documents in an Internet Explorer–only environment, you can let them select the level of detail that will be displayed by combining XML, XSLT, script, and methods of the Document Object Model.

The transformNode() method of the DOM, in particular, enables you to dynamically apply different XSL stylesheets to the same XML source document, producing different views of the same document at the click of a button.

The simple XML document in Listing 1 is structured into a hierarchy of topics with IBM's Darwin Information Typing Architecture (DITA), a DTD for creating modular documentation. (For more on DITA, see www-106.ibm.com/developerworks/xml/.) The document in Listing 1 can be displayed using a single XSLT stylesheet by embedding the xml:stylesheet processing instruction after the XML version processing instruction, as this code shows:

```
<?xml version="1.0" standalone="yes"?>
<?xml:stylesheet type="text/xsl" href=
 "listing2.xsl"?>
```

When an XML document with an xml:stylesheet processing instruction is loaded into Internet Explorer 5.0 or higher, it's formatted according to the template rules in the stylesheet. (Because the example stylesheets in this article conform to the W3C's XSLT Recommendation while the MSXML parser with which Internet Explorer 5 is natively equipped does not, you must have installed at least version 3 of the MSXML parser and be running it in replace mode for the code to work. You can obtain the latest version of the MSXML parser at http://msdn.microsoft.com and install it in a few minutes.)

Because the xml:stylesheet processing instruction can apply only one stylesheet to the document, it limits readers to the view prescribed by the stylesheet named listing3.xsl. The reader, however, may prefer to view only a certain level of the document's detail. Some readers, for instance, may want to read only an abstract, others an abridged view, and still others the entire piece. You can use a scripting language that supports the DOM to dynamically apply different stylesheets to an XML source document, thereby allowing readers to switch among various views of the same Web page with the press of a button.

A W3C specification, the DOM is a platform- and language-neutral interface that provides programs and scripts with the means to dynamically access and modify the content, structure, and style of documents, including those in HTML, XHTML, and XML format. For details, see the W3C's DOM page at www.w3.org/DOM/.

The transformNode Method

It's the DOM's transformNode method that lets you dynamically apply XSLT stylesheets to an XML document. If you create two instances of the DOM's Document object and load the XML source into one of them and the XSL stylesheet into the other, you can then use the stylesheet to transform the XML source by invoking the transformNode() method.

The syntax of the transformNode() method, which is available in ASP, JScript, Visual Basic, and VBScript, goes like this:

```
strResult = objXMLNode.transformNode
 (objStylesheet)
```

The objXMLNode typically takes a DOM Document object, though it can also take a node and its children within a DOM Document. The objStylesheet is usually a DOM Document instance containing a valid XSL stylesheet. The strResult, which is a string that contains the results of the XSLT transformation, can then be put into an HTML document by, for example, creating a <div id="results"></div> tag, setting a reference to it, and then using the innerHTML method to insert the string results.

Using TransformNode() to Switch Stylesheets

I've written two basic stylesheets for my source document:

- Listing 2 displays the first level of nested topics, giving readers a summary of sorts.
- Listing 3 displays the document's entire contents.

To enable a user with IE5 or higher running at least version 3 of the MSXML parser to select the level of detail of the document to be viewed, you'll first need to use script to load both the XML source file and the default XSLT stylesheet into the MSXML parser. The page will then have to provide users with a means to toggle between the different views while the script, when activated by a button pressed by the user, presents the document after it's transformed by the other stylesheet.

The script, which is embedded in an HTML page, appears in Listing 4, which can be downloaded at www.sys-con.com/xml/sourcec.cfm. A similar version of this script, replete with debugging logic, is available in Michael Kay's *XSLT Programmer's Reference* (Wrox), a book that covers XSLT in full and provides an excellent rundown of how to use XML and XSLT with Internet Explorer as well as an insightful discussion of the different versions of the MSXML parser.

After the HTML page in Listing 4 loads, the onLoad attribute of the <body> tag triggers the script's function and displays the contents of the document using the default stylesheet in Listing 2.

```
<body onLoad="transformPage
 ('listing2.xsl')">
```

The HTML page presents the user with two buttons, each of which specifies a different stylesheet to be used as the stylesheet parameter in the script's transformPage function without sending a request back to the server:

```
<button onClick=
 "transformPage('listing3.xsl')">
 View Entire Document</button>
```

```
<button onClick="transformPage
 ('listing2.xsl')">View Summary
 </button>
```

When a user selects one of the buttons, the stylesheet specified in the argument of transformPage function is applied to the XML document and the results are inserted inside the <div id="results"> tag.

This is just a sample of what can be done when you combine XML, XSLT, script, and methods of the DOM. If you're displaying complex documentation on the Web, such methods can be expanded to develop potent interfaces that give users the power to quickly find the information that suits their needs.

SHOENISH@RCN.COM

AUTHOR BIO

Steve Hoenisch, a technical writer/consultant with Verizon Wireless, is a former journalist and teacher. He has been developing Web sites since 1996.

LISTING 1

```
<?xml version="1.0"?>
<?xml:stylesheet type="text/xsl" href="listing2.xsl"?>
<!DOCTYPE topic SYSTEM "dtd/topic.dtd">
<topic><title>Separating Content from Presentation
 </title><body></body>
<topic>
 <title>Introduction</title>
 <body>
  <p>You've probably heard the propaganda by now: XML blesses
    you with a way to separate content from presentation.
    Separation in turn yields productive gains over HTML and
    other data formats used to manage content. What makes
    all this possible? XSLT.
</p>

 </body>

</topic>
<topic><title>XSLT</title><body><p>Extensible Stylesheet
 Language for Transformations, or XSLT, is a functional
 programming language that enables us to bridge the gap
 between content and presentation by providing a means of
 specifying how a content-based XML document is transformed
 into a presentation-oriented document or data format.
 </p></body><topic>
 <title>XSL History</title>
 <body>
  <p>The birth of XSLT breaks down like this: Extensible
   Stylesheet Language (XSL) was conceived as an XML
   application for expressing stylesheets capable of
   manipulating XML documents. After its submission to the W3C
   in 1997, XSL split into two stylesheet standards:
   Extensible Stylesheet Language for Transformations (XSLT)
   and Extensible Stylesheet Language Formatting Objects
   (XSL-FO). </p>

 </body>
<topic><title>XPath</title><body><p>The early XSL proposals
 also gave rise to another breakaway standard, XPath. It
 grew out of the location-finding aspects of the XSL and
 XPointer specifications, which had been independently using
 similar mechanisms to find information in XML documents.
 XSLT relies on XPath to locate specific nodes or node sets
 within XML documents. </p>
  </body></topic><topic><title>CSS</title><body><p>XSL has
   another related standard, though unlike XSL it is not
   based in XML: Cascading Style Sheets (CSS). In the
   context of web publishing, Cascading Style Sheets (CSS)
   can also be used, at least theoretically, to statically
   format XML documents for display, but it cannot be used
   to transform them in any meaningful way.</p>
  </body></topic></topic></topic></topic>
```

LISTING 2

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"
                 xmlns:xsl=
                  "http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" encoding="iso-8859-1"
             indent="yes" />
<xsl:template match="/">
<html>
  <head>
  <style type="text/css">
.titleHead { font-family: helvetica, arial, sans-serif;
 font-size: 160%; font-weight: bold; color: #804000; }
</style>
<!--  <link rel="stylesheet" type="text/css" href=
 "ss/mystyles.css" />  -->
<meta http-equiv="author" content="Steve Hoenisch"/>
<title><xsl:value-of select="topic/title" /></title>
</head>
<body class="base">
<span class="titleHead"><xsl:value-of select=
 "topic/title" /><br/></span>
<hr size="1" width="100%" color="brown"/>
<xsl:apply-templates select="topic/topic"/>
</body>
</html>
</xsl:template>

<xsl:template match="topic">
<h1><xsl:value-of select="title" /></h1>
<p><xsl:value-of select="body/p" /></p>
</xsl:template>

</xsl:stylesheet>
```

LISTING 3

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"
                 xmlns:xsl=
                  "http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" encoding="iso-8859-1"
             indent="yes" />
<xsl:template match="topic/title">
<html>
  <head>
  <style type="text/css">
.titleHead { font-family: helvetica, arial, sans-serif;
 font-size: 160%; font-weight: bold; color: #804000; }
</style>
<!--  <link rel="stylesheet" type="text/css" href=
 "ss/mystyles.css" />  -->
<meta http-equiv="author" content="Steve Hoenisch"/>
<title><xsl:value-of select="." /></title>
</head>
<body class="base">
<div class="content_target">
<span class="titleHead"><xsl:value-of select="."
/><br/></span>
<hr size="1" width="100%" color="brown"/>
<xsl:apply-templates select="topic"/>
</div>
</body>
</html>
</xsl:template>

<xsl:template match="topic/topic/title">
<h1><xsl:value-of select="." /></h1>
</xsl:template>

<xsl:template match="topic/topic/topic/title">
<h2><xsl:value-of select="." /></h2>
</xsl:template>

<xsl:template match="topic/topic/topic/topic/title">
<h3><xsl:value-of select="." /></h3>
</xsl:template>

<xsl:template match="topic/topic/topic/topic/topic/title">
<h4><xsl:value-of select="." /></h4>
</xsl:template>

<xsl:template match="topic/topic/topic/topic/topic/topic/title">
<h5><xsl:value-of select="." /></h5>
</xsl:template>

<xsl:template match="p">
<p><xsl:value-of select="." /></p>
</xsl:template>

</xsl:stylesheet>
```

Download the Code
www.sys-con.com/xml

WebServices JOURNAL .NET J2EE XML

XML JOURNAL

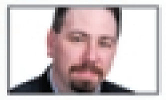
THE FIRST & ONLY WEB SERVICES RESOURCE CD!

WEB SERVICES RESOURCE CD

THE SECRETS OF THE WEB SERVICES MASTERS

INCLUDES EXCLUSIVE .NET ARTICLES

MORE THAN 400 EXCLUSIVE WEB SERVICES & XML ARTICLES



EDITED BY SEAN RHODY

$119 CD VALUE FROM WEB SERVICES JOURNAL

Web Services EDGE conference & expo $100 coupon inside

EVERY ISSUE OF WSJ & XML-J EVER PUBLISHED

THE MOST COMPLETE LIBRARY OF EXCLUSIVE WSJ & XML-J ARTICLES ON ONE CD!

"The Secrets of the Web Services Masters"

CD is edited by well-known WSJ Editor-in-Chief Sean Rhody and organized into more than 40 chapters containing more than 400 exclusive WSJ & XML-J articles.

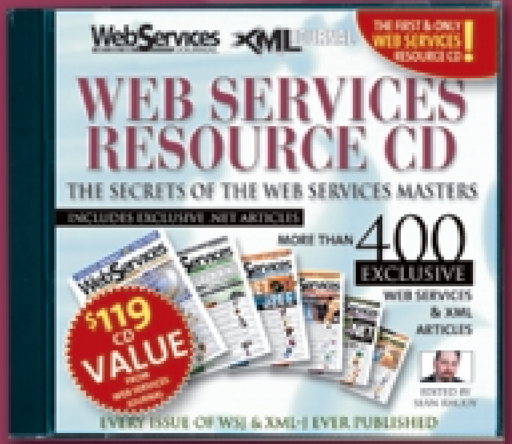
Easy-to-navigate HTML format!

Bonus:

Full .PDF versions of every WSJ & XML-J published since the first issue

XML in Transit	XML Industry Insider	XML & Databases	UML
XML B2B	<e-BizML>	Electronic Data Interchange	Integration
Java & XML	XML & Business	Ubiquitous Computing	WSDL
The XML Files	XML Demystified	Information Management	Beginning Web Services
XML & WML	XML & E-Commerce	Objects & XML	Web Services Tips & Techniques
Voice XML	XML Middleware	XML Pros & Cons	Frameworks
SYS-CON Radio	XML Modeling	Java Servlets	Management
XML & XSLT	CORBA & XML	XML Filter	Security
XML & XSL	Data Transition		UDDI
XML & XHTML	XML @ Work		.NET
2B or Not 2B			
XML Script			

3 YEARS
25 ISSUES
400 ARTICLES
ONE CD



Now Shipping

$79

ONLINE ORDER AT JDJSTORE.COM SAVE $40

The World Wide Web Consortium is working on finalizing the specification for XQuery, aiming for a final release in mid to late 2002. XQuery is a powerful and convenient language that's designed for processing XML data – not just files in XML format, but other data (including databases) whose structure (nested named trees with attributes) is similar to XML.

XQuery is an interesting language with some unusual ideas. This article is intended to give you a hawk's-eye view of XQuery, introducing the main ideas you should understand before you go deeper…or actually try to use it!

Written by Per Bothner

What Is XQuery

When you need a powerful and convenient tool for analyzing or generating XML…

An Expression Language

The first thing to notice is that in XQuery everything is an expression that evaluates to a value. An XQuery program or script is just an expression, together with (optionally) some function and other definitions. So the following:

```
3+4
```

is a complete and valid XQuery program that evaluates to the integer 7.

There are no side effects or updates in the XQuery standard, though those will probably be added at a future date. The standard specifies the result value of an expression or program, but it doesn't specify how it's to be evaluated. Therefore an implementation has considerable freedom in how it evaluates an XQuery program and what optimizations it does.

Here is a conditional expression that evaluates to a string:

```
if (3 < 4) then "yes!" else "no!"
```

You can define local variable definitions using a let-expression:

```
let $x := 5 let $y := 6 return 10*$x+$y
```

This evaluates to 56.

Primitive Data Types

The primitive data types in XQuery are the same as for XML Schema:

- Numbers, including integers and floating-point numbers
- The boolean values true and false
- Strings of characters (for example, "Hello world!" [these are immutable, i.e., you can't modify a character in a string])
- Various types to represent dates, times, and durations
- A few XML-related types (for example, a QName is a pair consisting of a local name [like template] and a URL, which is used to represent a tag name like xsl:template after it has been namespace-resolved)

Derived types are variations or restrictions of other types, for example, range types. Primitive types and the types derived from them are known as *atomic types*, because an atomic value doesn't contain other values. Thus a string is considered atomic because XQuery doesn't have character values.

Node Values and Expressions

XQuery, of course, also has the necessary data types needed to represent XML values. It does this using *node* values, of which there are seven kinds: element, attribute, namespace, text, comment, processing-instruction, and document (root) nodes. These are very similar to the corresponding DOM classes such as Node, Element, and so on. Some XQuery implementations use DOM objects to implement node values, though implementations may use other representations.

Various standard XQuery functions create or return nodes. For example, the document function reads an XML file specified by a URL argument, and returns a document root node. (The root element is a child of the root node.)

You can also create new node objects directly in the program. The most convenient way to do that is to use an *element constructor* expression, which looks just like regular XML data:

```
<p>See <a href="index.html"><i>here</i></a> for info.</p>
```

You can use {*curly braces*} to embed an XQuery expression inside element constructors:

```
let $i := 2 return
let $r := <em>Value </em> return
  <p>{$r} of 10*{$i} is {10*$i}.</p>
```

creates:

```
<p><em>Value </em> of 10*2 is 20.</p>
```

Popular *template processors* such as JSP, ASP, and PHP allow you to embed expressions in a programming language into HTML content. XQuery gives you that ability, plus the ability to embed XML/HTML forms inside expressions and to have them be the value of variables and parameters.

XQuery node values are immutable (you can't modify a node after it has been created).

Sequences

We've seen atomic values (numbers, strings, etc.) and node values (elements, attributes, etc). These are together known as *simple values*. XQuery expressions actually evaluate to *sequences* of simple values. The comma operator can be used to concatenate two values or sequences. For example:

```
3,4,5
```

is a sequence consisting of three integers. Note that a sequence containing just a single value is the same as that value by itself, and you cannot nest sequences. To illustrate this, we'll use the count function, which takes one argument and returns the number of values in that sequence. So this expression:

```
let $a := 3,4
let $b := ($a, $a)
let $c := 99
let $d := ()
return (count($a), count($b), count($c), count($d))
```

evaluates to (2, 4, 1, 0), because $b is the same as (3,4,3,4).

Many of the standard functions for working with nodes return sequences. For example, the children function returns a sequence of the child nodes of the argument:

```
children(<p>This is <em>very</em> cool.</p>)
```

and returns this sequence of three values:

```
"This is ", <em>very</em>, " cool."
```

Path Expressions and Relationship to XPath

XQuery borrows *path expressions* from XPath. In fact, XQuery can be viewed as a generalization of XPath: except for some obscure forms (mostly unusual "axis specifiers"), all XPath expressions are also XQuery expressions. For this reason the XPath specification is also being revised by the XQuery committee, with the plan that XQuery 1.0 and XPath 2.0 will be released about the same time.

The following simple example assumes an XML file "mybook.xml" whose root element is a <book>, and it contains some <chapter> children:

```
let $book := document("mybook.xml")/book
return $book/chapter
```

The document function returns the root node of a document. The /book expression selects the child elements of the root that are named book, so $book gets set to the single root element. The $book/chapter selects the child elements of the top-level book elements, which results in a sequence of the second-level chapter nodes, in *document order*.

The next element includes a *predicate*.

```
$book//para[@class="warning"]
```

The double slash is a convenience syntax to select all descendants (rather than just children) of $book, selecting only <para> element nodes that have an attribute node named class whose value is "warning".

One difference to note between XPath and XQuery is that XPath expressions may return a *node set*, whereas the same XQuery expression will return a node sequence. For compatibility, these sequences will be in *document order* and with duplicates removed, which makes them equivalent to sets.

By the way, XPath expressions are mostly used as patterns in XSLT *stylesheets*. XSLT (XSL Transformation, where XSL stands for XML Stylesheet Language) is a rule-based language for transforming an input XML document into a result XML document. XSLT is very useful for expressing very simple transformations, but more complicated stylesheets (especially anything with nontrivial logic or programming) can often be written more compactly and readably using XQuery.

My article, "Generating XML and HTML Using XQuery" (www.gnu.org/software/qexo/XQ-Gen-XML.html), explains further how to generate XML documents and HTML Web pages using XQuery.

Iterating over Sequences

A *for* expression lets you "loop" over the elements of a sequence:

```
for $x in (1 to 3) return $x,10+$x
```

The for expression first evaluates the expression following the *in*. For each value of the resulting sequence, the variable (in this case $x) is then bound to the value, and the return expression evaluated using that variable binding. The value of the entire for expression is the concatenation of all values of the return expression, in order. So the example evaluates to this six-element sequence:

```
1,11,2,12,3,13
```

Here's a more useful example. Assume again that mybook.xml is a <book> that contains some <chapter> elements. Each <chapter> has a <title>. The following will create a simple Web page that just lists the titles:

```
<html>{
  let $book := document("mybook.xml")/book
  for $ch in $book/chapter
    return <h2>{$ch/title)</h2>
}</html>
```

The term *FLWR expressions* includes both for and let expressions. The acronym FLWR refers to the fact that it consists of one or more *f*or and/or *l*et clauses, an optional *w*here clause, and a *r*esult clause. A where clause causes the result clause to be evaluated only when the where expression is true.

The following is an example illustrating the where clause. This example has a nested loop, allowing us to combine two sequences, one of customer elements, the other of order elements. We want to find the name(s) of customers who have ordered the part whose part_id is "xx".

```
for $c in customers
for $o in orders
where $c.cust_id=$o.cust_id and $o.part_id="xx"
return $c.name
```

This is essentially a *join* of two tables, as commonly performed using relational databases. An important goal for XQuery is that it should be usable as a query language for "XML databases." Compare the corresponding SQL statement:

```
select customers.name
from customers, orders
where customers.cust_id=orders.cust_id
  and orders.part_id="xx"
```

Functions

XQuery wouldn't be much of a programming language without user-defined functions. Such function definitions appear in the *query prologue* of an XQuery program. It's worth noting that function parameters and function results can be primitive values, nodes, or sequences of either.

Below we define a recursive utility function. It returns all the descendant nodes of the argument, including the argument node itself. It does a depth-first traversal of the argument, returning the argument and then looping over the argument node's children, recursively calling itself for each child.

```
define function descendant-or-self ($x)
{
  $x,
  for $y in children($x)
    return descendant-or-self($y)
}
descendant-or-self(<a>X<b>Y</b></a>)
```

evaluates to this sequence of length 4:

```
<a>X<b>Y</b></a>; "X"; <b>Y</b>; "Y"
```

Sorting and Context

If you want to sort a sequence, you can use a sortby expression. For example, to sort a sequence of books in order of author name you can do:

```
$books sortby (author/name)
```

The sortby takes an input sequence (in this case $books) and one or more *ordering expressions*. During sorting the implementation

Various standard XQuery functions create or return nodes. For example, the document function reads an XML file specified by a URL argument, and returns a document root node needs to compare two values from the input sequence to determine which comes first. It does that by evaluating the ordering expression(s) in the *context* of a value from the input sequence. So the path expression author/name is evaluated many times, each time relative to a different book as the *context* (or *current*) item.

Path expressions also use and set the context. For example, in author/name the name children that are returned are those of the context item, which is an author item.

Type Specification

XQuery is a strongly typed programming language. Like Java, C#, and other languages, it is a mix of static typing (type consistency checked at compile time) and dynamic typing (runtime type tests). However, the types in XQuery are different from the classes familiar from object-oriented programming. XQuery has types to match its data model, and allows you to import types from XML Schema.

```
if ($child instance of element section)
then process-section($child)
else ( ) {--nothing--}
```

This invokes the process-section function if the value of $child is an element whose tag name is section. XQuery has a convenient typeswitch shorthand for matching a value against a number of types. Here's an example of converting a set of tag names to a different set. It's a simple example of the kind of transformations that XSLT does well.

```
define function convert($x) {
  typeswitch ($x)
    case element para return <p>
     {process-children($x)}</p>
    case element emph  return <em>
     {process-children($x)}</em>
    default return process-children($x)
}
define function process-children($x) {
  for $ch in children($x) return convert($ch)
}
```

Resources

The primary XQuery resource is www.w3.org/XML/Query at the World Wide Web Consortium. This has links to the draft standards, mailing lists, and implementations. The main documents follow:

- The actual XQuery specification (www.w3.org/TR/xquery) isn't difficult to read, and is probably where you should go next.
- The Data Model specification (www.w3.org/TR/query-datamodel) goes into nodes and the functions you can use to manipulate them.
- The Functions and Operators specification (www.w3.org/TR/xquery-operators) defines the other (nonnode) functions, including string and date/time functions.
- The Use Cases document (www.w3.org/TR/xmlquery-use-cases) contains lots of useful examples of XQuery programs to solve specific problems.
- The Formal Semantics specification (www.w3.org/TR/query-semantics) uses formal mathematical notation and is not for the faint of heart. Most people should skip it.

PER@BOTHNER.COM

AUTHOR BIO

In 1996 Per Bothner started the GCJ (GNU compiler for the Java platform) project, which is the most active open-source Java implementation project. He implemented the Kawa framework for compiling high-level languages (including Scheme and XQuery) to Java bytecodes. He is currently an independent consultant.

WHERE TO GO NEXT

There aren't many books on XQuery yet, mainly because there are significant loose ends in the specification. At the time of this writing, there is one book: *Early Adopter XQuery* from Wrox. I'm working with other authors on a book for Sams, which we hope will be ready soon after the standard is finalized.

Obviously, there are no complete standards-conforming implementations either, but the XQuery site lists known implementations, some of which have executable demos. The only open-source implementation currently available seems to be my own Qexo implementation (see www.gnu.org/software/qexo/). The Qexo implementation is interesting in that it compiles XQuery programs on-the-fly directly to Java bytecodes. I welcome you to experiment with it. But in any case, I do recommend considering XQuery when you need a powerful and convenient tool for analyzing or generating XML.

XQuery wouldn't be much of a programming language without user-defined functions. Such function definitions appear in the query prologue of an XQuery program

WRITTEN BY **HITESH SETH**

Selecting a VoiceXML Gateway

Here are the criteria to help you

As an open standard, VoiceXML truly leverages the knowledge and products that the industry has developed around Advanced Speech Recognition (ASR), Text-to-Speech (TTS), telephony interfaces, voice printing, and Voice over IP (VoIP). As an open standard, what VoiceXML has made possible is flexible models for development and deployment.

Some of my previous articles reviewed deployment tools that are available for developers to rapidly build and test VoiceXML-based interactive speech applications. In this issue I'll preview some options that are available to enterprise speech recognition developers and IT managers for deploying a VoiceXML application.

In the earlier articles we discovered that a number of third-party vendors provide hosted VoiceXML development tools and another group provides desktop-based development tools. Depending on your scenario and your investment in speech recognition, you can pick and choose which development tool model suits your environment. And once you're ready to deploy VoiceXML applications, a number of similar models are available. Although it's definitely possible to buy the entire suite of speech recognition and telephony infrastructure in-house – if you have the investment capability – it's also possible to lease the whole infrastructure, including the telephony lines and integration, from an outsourced service provider.

What Is a VoiceXML Gateway?

As depicted in Figure 1, a VoiceXML gateway is the key link between the telephony infrastructure and your VoiceXML application, representing a suite of integrated technologies. Typically, a gateway includes technology components such as ASR, TTS, VoiceXML interpreter, and telephony integration. Optionally, it can also include voice authentication/voice-print technology, a set of platform extensions to VoiceXML, and reusable components. Even though I've represented a VoiceXML gateway conceptually as a single piece of infrastructure, it's important to understand that it's typically an integration of multiple technologies. As we'll explore in the rest of the article, some vendors sell these technologies both individually and as an integrated suite of products.

Selecting the Gateway

Developing and deploying speech applications is a challenging task. Typically, developing and deploying an IVR application has meant investing in a set of proprietary technologies and systems hosted either within your own company or as an expensive outsourced offering. VoiceXML has opened up a landscape of speech application deployment to a whole suite of third-party technology providers. A number of vendors have brought in interesting tools and application models to support rapid development and deployment of VoiceXML applications. In a nutshell, three approaches to deploying your VoiceXML applications to the public telephony network have emerged, based on how you set up your key infrastructure link, the VoiceXML gateway:

- ***Build:*** Build your own VoiceXML gateway by integrating a suite of best-of-breed speech recognition technologies. Connect this gateway with the PSTN (public switched telephony network) or an internal VoIP network (if using the application within corporate boundaries).
- ***Buy:*** Buy an integrated VoiceXML gateway. You still need to connect it with the telephony network.
- ***Rent:*** Outsource the functions of the VoiceXML gateway and integration with the telephony network to a service provider. You still have to develop and host your VoiceXML application, though. Of course, you can still utilize your regular Web hosting provider for hosting/colocating your VoiceXML application.

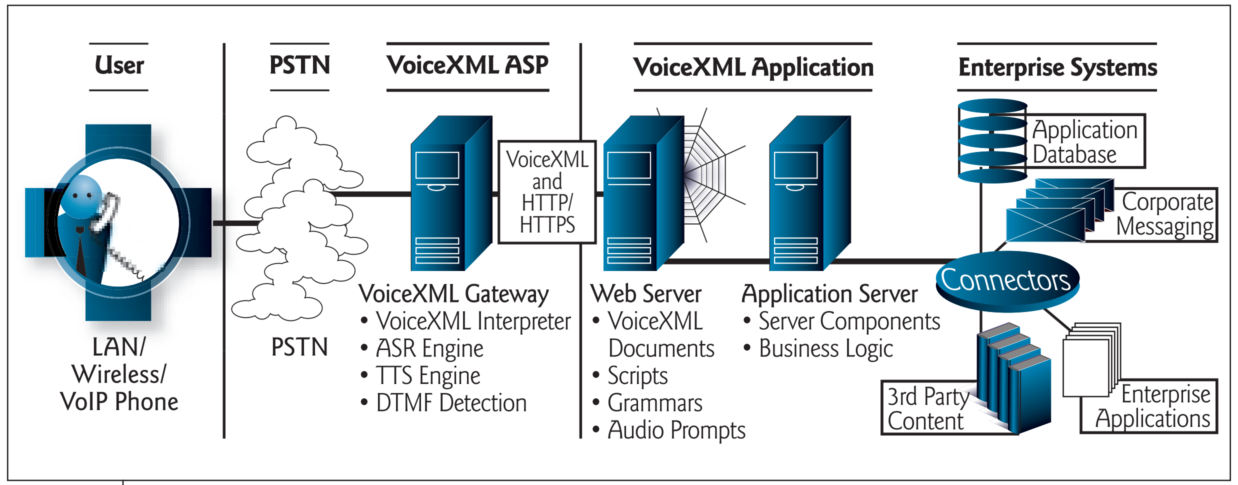


FIGURE 1 | VoiceXML application model

AUTHOR BIO

Hitesh Seth is chief technology evangelist for Silverline Technologies, a global e-business and integration services firm. His technical interests include enterprise architecture and design using frameworks such as J2EE and .NET, XML, wireless computing, speech applications, Web services, enterprise portals, and enterprise integration.

Key Selection Criteria

Following are the key selection criteria you should analyze while evaluating the various alternatives:

- VoiceXML 1.0/2.0 compliance
- Stability of the solution
  - *— Availability*
  - *—Vendor presence in your industry*
- Number of concurrent users supported
  - *—Typical versus spike*
- Grammar formats supported
- Support for integrated development tools
- Integration with existing call center systems
- ASR engines supported
- TTS engines supported
- Languages supported
- Audio formats supported (for prerecorded audio)
- Reusable components
  - *—Reusable dialogs*
  - *—Prebuilt library of audio prompts*
  - *—Prebuilt library of grammars*
- Extensive auditing/debugging capability
- Density (number of ports supported per server)
- Investment required (different pricing models per port, per minute usage, etc.)
  - *—Onetime investment required*
  - *—Recurring expense*

The remainder of this article is a critical analysis of the three approaches for deploying VoiceXML applications. The objective is to provide you with the knowledge to make the right decision for your next VoiceXML-based application.

Build your own VoiceXML gateway

This approach requires you to integrate the best-of-breed speech recognition technologies, hardware, and telephony platforms and create your own representation of a VoiceXML gateway. Typically, you'd purchase a server (Unix or Windows based), buy telephony integration boards (such as those from Intel Dialogic), and install other components on the same server or, if supported by the vendors, in a distributed scenario. Table 1 lists some key characteristics of the "build" solution.

Buy integrated VoiceXML gateway

The major difference between this and the build approach is that you outsource the integration of the suite of speech recognition and text-to-speech technologies to a third-party vendor. Although you'd purchase an integrated VoiceXML gateway from a third-party vendor, you'd still have to integrate it with the telephony infrastructure (lease lines, etc.). Table 2 lists key characteristics of the "buy" solution.

Rent outsourced VoiceXML gateway

The "rent" model, also known as a Voice ASP (application service provider) or sometimes a VSP (voice service provider), provides the capabilities of a VoiceXML gateway, including speech recognition, TTS, integration with the telephony system, and so on, in a completely outsourced manner. Typically, the VoiceXML application would be developed on your favorite Web/application server platform and, through either a public Internet or a virtual private network, would link the HTTP/HTTPS–based speech application to a telephone number(s). Table 3 lists some key characteristics of the "rent" solution.

Conclusion

If your company uses an IVR system/technology today, chances are that your IVR provider is already working on being VoiceXML compliant. It's also possible that some components of your existing IVR infrastructure (e.g., speech recognition/ASR, telephony boards, or TTS) can be upgraded/utilized as essential components of a VoiceXML gateway. In these scenarios a build approach may be more suitable.

However, if your company doesn't have a lot of experience working with the speech recognition/telephony infrastructure, and lowering the cost of initial investment is critical, you'd probably like to test the waters with an outsourced voice service provider. Most such providers provide a free development capability that would allow you to build a prototype and show it to some key users to get feedback. Ultimately you have to consider the criteria that are most critical for your environment and make the build/buy/rent decision for deploying your VoiceXML application.

HKS@HITESHSETH.COM

Highlights	• Best-of-breed approach • Combine your favorite ASR/TTS/VoiceXML interpreter • Buy telephony integration boards • Lease telephony lines *Integrate them together*
Pros	• Best-of-breed integration • You get to select the best technologies • You can use existing investments in individual components • Low cost for high-volume scenarios
Cons	• Complexity of integration & operations
Partial List of Vendors	• *ASR:* Nuance, SpeechWorks, etc. • *TTS:* Nuance, SpeechWorks, Fonix, AT&T, ScanSoft • *Telephony:* Intel Dialogic, Cisco • *VoiceXML Interpreter:* Nuance Voice Web server, SpeechWorks OpenSpeech browser, OpenVXL, IBM, etc.

TABLE 1 | Key characteristics of the "build" solution

Highlights	• Buy an out-of-box integrated VoiceXML gateway with all the key technologies • Lease telephony lines *Integrate them together*
Pros	• Partial best-of-breed integration • Lowers complexity of integration • Low cost for high-volume scenarios
Cons	• Complexity of operations
Partial List of Vendors	• Cambridge voice gateway • GetVocal VoiceXML gateway • IBM • InterVoice-Brite • Lucent speech server • Motorola • Nortel Networks • Verascape • VoiceGenie

TABLE 2 | Key characteristics of the "buy" solution

Highlights	• Rent a VoiceXML gateway from a Voice ASP • Pay by the minute charges for your callers
Pros	• Offload speech technology complexity to an outsourced provider • No upfront capital investment • You can test the waters before investing in infrastructure
Cons	• High cost for high-volume scenarios • Lack of integration with internal systems (such as call centers, etc.) • Higher security concerns
Partial List of Vendors	• BeVocal • HeyAnita • iBasis • InterVoice-Brite • Qwest • Tellme • VoiceGenie • Voxeo

TABLE 3 | Key characteristics of the "rent" solution

Bob Hendry, a Java instructor at the Illinois Institute of Technology, is the author of Java as a First Language.

REVIEWED BY **BOB HENDRY**

OMNIOPERA

BY KAMIAK

KAMIAK
Box 151
Sheridan, WY 82801
Phone: 603 673-5163
E-Mail: info@kamiak.com
Web: www.kamiak.com

TEST ENVIRONMENT
OS: Microsoft NT4
Processor: Pentium III, 550MHz
Memory: 64MB

SPECIFICATIONS
Platforms: Platform independent
Pricing: Variously priced at $95 and $225 (see Web site for details)

I've been in the XML world for about two years now – a dinosaur at today's rate. Still, I have problems with understanding conceptual and syntax issues of WSDL and XML Schema – especially the latter. In addition to being complex, XML Schema and WSDL are relatively new (or soon to be new) WC3 recommendations. So now I have two WC3 recommendations that are intimidating and complex in nature. If they're difficult for me, what about XML beginners on my team? How will they understand them? Enough said…

What this industry really needs is an inexpensive tool that will allow the creation and modification of WSDL and XML schemas. This tool should allow an organization to author these documents and write its own Web service contracts, all while hiding the inner workings of WSDL and XML Schema. Basically, it should leave the creation and modification of the document to a higher level while it performs the grunt work. Then, with most of the code generated, it should allow detailed modification. I always like a tool that implements the famous 80/20 rule. Give me 80% of the functionality and let me worry about the rest. Kamiak Corporation's Omniopera fits this description. Most of the guesswork is removed. With only a few clicks your project is well on its way.

Kamiak, a software development tools company, specializes in Web service–related technologies. Omniopera is a WSDL and XML Schema editor. The philosophy behind it is that you can use WSDL as IDL to define a Web service interface up front, prior to Web service development. It's a great front end for any of the major Web service development tools such as Microsoft Visual Studio .NET, Borland Delphi 6, or Apache Axis.

Without Omniopera (or similar IDE), developers start with a development tool and create code to define their Web service. The development tool creates the WSDL to define the Web service interface. Machine-generated WSDL can result in a poorly defined interface, causing problems when people try to develop clients that work with the Web service. Mark Young, vice president at Kamiak, adds: "One of the things we've been emphasizing lately is that our users can start with industry-standard schemas (like those from ebXML) and develop their WSDL accordingly. We think it is much harder to do this without a tool like Omniopera, especially if a developer starts coding first." Omniopera is a solid tool in this area with useful extras.

Downloading was a snap – get a trial copy at www.omniopera.com. The setup file was small (4.8MG) and took only a few minutes to come across the wire. No surprises with the installation – just agree to a 10-day trial license and you're off and running. I'd like to thank Kamiak for not asking the name of my firstborn to download Omniopera. Rather than a long complicated questionnaire, Kamiak only wanted my e-mail address. Nice touch. The help system is also very clear and detailed.

At the heart of Omniopera is its graphical drag-and-drop interface (see Figure 1). This GUI allows trouble-free XML schema and WSDL editing, both performed in a treeview. The hierarchical structure of an XML document makes it a natural for a treeview.

Once you're equipped with Omniopera, look forward to taking tighter control of your Web services. I definitely found myself creating better contracts. In addition, with tighter data typing, I expect to spend less time explaining my Web services to other developers. Although I develop Web services on J2EE, I don't expect that I'd expend a whole lot of effort adapting my services to work with other platforms – like .NET.

All in all, Omniopera allows you to spend less time figuring out the precise source code incantation required to expose the sophisticated data structures that your Web service requires; your framework will automatically create this code for you. This all adds up to a tremendous time saving. Now *that* I can use!

BOB@SYS-CON.COM

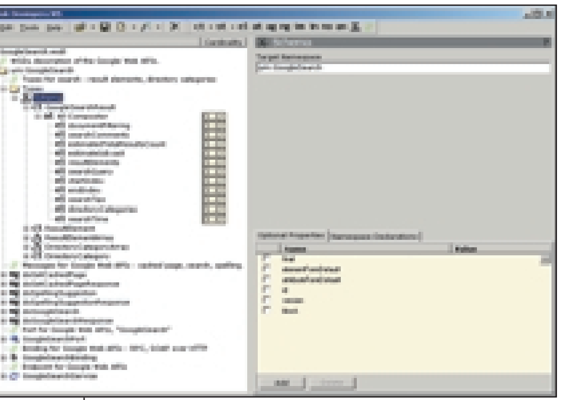


FIGURE 1 Omniopera's drag-and-drop interface

XML-J ADVERTISERINDEX

ADVERTISER	URL	PHONE	PAGE
ADOS Co., Ltd.	http://www.a-dos.com	81-3-5475-1551	59
Altova	www.xmlspy.com		60
CTIA Wireless I.T. & Internet 2002	www.ctiashow.com		13
engindata Research	www.engindata.com	201-802-3082	30,31
HitSoftware	www.hitsw.com/xmlrdb	408-345-4001	6
IBM	www.ibm/com/websphere/winning		2,3
JDJ Store	www.jdjstore.com	888-303-JAVA	47
Macromedia	www.macromedia.com/go/cfmxad	888-939-2545	9
Muscular Dystrophy Association	www.als.mdausa.org	800-572-1717	15
Simplex Knowledge	www.skc.com	845-620-3700	35
Sprint	http://developer.sprintpcs.com		4
SYS-CON Media	www.sys-con.com/suboffer.cfm	888-303-5282	33
SYS-CON Media	www.sys-con.com	888-303-5282	45
Web Services Resource CD	www.jdjstore.com	888-303-JAVA	40, 41
WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	29
WebServices Edge World Tour	www.sys-con.com	201-802-3069	36, 37
WebSphere Developer's Journal	www.sys-con.com/websphere	888-303-5282	23

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of ***XML-Journal***. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in ***XML-Journal***. Advertisements are to be printed at the discretion of the Publisher. This discretion includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc.

WRITTEN BY **JOHN EVDEMON**

The Importance of Integration Standards

Recent developments in XML-based integration

On a recent trip overseas I neglected to pack the adapter plugs that enable you to plug an electrical cord from one country into an outlet in another. If you travel overseas you soon realize that many countries have incompatible electrical outlets.

Outlets in the UK and Ireland use a plug with three thick, flat pins, while Australia and China use two flat, angled pins. In addition to the hardware incompatibilities, voltage standards tend to vary by country – U.S. appliances function on 120 volts AC while many foreign power sources provide 240 volts AC. Plugging a U.S. appliance into a foreign outlet (assuming you remembered your adapter plugs) can damage the appliance unless a voltage converter is used (most laptop power supplies include a built-in voltage converter). My laptop battery was depleted and my U.S. plug (three pins: two flat and one circular) could not be used with a Korean socket (two circular receptacles). As I watched my laptop die a slow, painful death (ironically begging me to switch to another power source as soon as possible), I reflected on how much we take standardization for granted.

A Sony CD player and JVC cassette deck can be quickly and easily connected to a Panasonic audio receiver. A CD from a small label like Skunk Records sounds just as good as a new CD from BMG. Firefighters no longer need to worry about hose fittings that vary from one hydrant to the next. USB's token, data, and handshake protocols enable us to quickly and easily connect a broad range of devices to our computers. TCP/IP has become largely ubiquitous, enabling computers (and other devices) to be quickly and easily assembled into large networks. The magazine you're now reading would not have been possible without standards regarding printing, binding, and distribution.

Clearly, without standardization there would be no hope for mass production or mass communication. The so-called "new economy" wouldn't have had a chance because the "old economy" would have sputtered and died (much like my poor laptop). Although hundreds of thousands of standards are available, those pertaining to integration are still maturing. Let's take a quick look at some of them.

JCA

Much like my missing adapter plugs, integration vendors provide a broad range of system adapters for connecting to legacy platforms, key business applications, and trading partners. System adapters enable integration platform providers to tie together disparate systems in a fairly seamless manner using both synchronous and asynchronous communications. Unlike my missing adapter plugs, integration adapters aren't plug-and-play. One vendor's integration adapters are not compatible with another's (the features of a given adapter also vary from one vendor to another).

Some efforts are under way to resolve this issue – most notably the J2EE Connector Architecture. The JCA is expected to help large platform vendors move into the lucrative EAI market (currently dominated by smaller, more specialized companies). While JCA may make it easier to reuse integration adapters, it is very immature (released in July 2001). It also lacks many of the features expected in vendor-specific system adapters (such as asynchronous communications, message brokering, and process flow). Many platform vendors have witnessed a rapid commoditization of their products; advances in the J2EE platform have made it increasingly difficult to provide value-added services unique to a given vendor. The consolidation of the EAI market that started with IBM's purchase of Crossworlds and Sybase's purchase of Neon will undoubtedly continue over the next year or two.

XML Messaging Formats and XML Syntax

At four years old, the XML Technical Recommendation can be thought of as a precocious preschooler: wonderful to work with but a source of endless headaches and stress if not properly managed. While XML was originally designed for lightweight content management, integrators soon recognized that it provided a technique for "universal data serialization," enabling an open, portable abstraction layer between the message and the systems designed to create and consume the message.

SOAP and XML-RPC are two popular XML-based messaging initiatives that describe enveloping and message handling, but place few restrictions on the message itself. SOAP and XML-RPC are basically remote procedure calls (RPC) that use an XML framework. SOAP is supported by most integration vendors and has evolved into a de facto standard for lightweight application messaging protocols.

While SOAP has become a recognized leader for defining XML-based message envelopes, standards regarding the actual payload continue to evolve. Initiatives such as OAG, RosettaNet, XCBL, and UBL all offer XML representations of common business transactions (e.g., purchase orders, invoices). A leader in this space has yet to emerge, causing confusion for many firms moving to embrace XML. The lack of a prominent standard may be causing a bit of a backlash against XML. The General Accounting Office recently issued a report that identified several risks associated with XML, some of which are listed below:

- Standards for defining business transactions are incomplete.
- Standards for identifying and creating legally binding trade agreements are incomplete.
- The lack of standards will cause a proliferation of redundant data structures and vocabularies.
- Security issues associated with XML and XML messaging (e.g., SOAP) are incomplete.

While these criticisms sound quite harsh, they are also quite true. Luckily, several working groups (ebXML, W3C, X12, OASIS, and others) are working to address these shortcomings.

ebXML is defining CPP and CPA, standards designed to create and bind trade agreements. OASIS is working on UBL, the Universal Business Language (based largely on ebXML Core Components). The W3C's Web services initiative is working to improve XML protocols (such as SOAP) to add security, transaction support, and other features. X12, the standards body behind EDI in the U.S., is developing requirements for an XML representation of EDI. While the X12 requirements are still under development, the initial schemas do not (yet) resemble any of the previously mentioned initiatives (strangely enough, the schemas don't seem to reflect any of the structures or naming conventions defined by EDI either).

The XML Technical Recommendation (the document that defines the syntax of XML) is also evolving. The original recommendation was released in February 1998, with an update issued in October 2000 to correct various errata. XML 1.1 (code named "Blueberry") is currently being developed by the W3C XML Core Syntax Working Group. XML 1.1 has three distinct objectives:

1. Ensure XML compliance with Unicode 3.0.
2. Add support for NEL (Network End-point Locator), the new line character used on IBM mainframes.
3. Decouple XML dependence from specific Unicode releases (thereby avoiding the need for new XML versions with each new Unicode release).

These objectives are a bit controversial for many people in the XML community. The second objective is particularly contentious since it appears to be a vendor-centric concession. In reality, much of the world's data continues to reside on mainframes; lowering the barriers to mainframe XML compliance ensures that porting this data to an XML syntax will be relatively painless.

Regardless of the controversy, the 1.1 Working Draft inspired people to reconsider the original XML 1.0 Technical Recommendation. Many people thought the XML 1.1 Working Draft should have introduced additional changes, such as the inclusion of XML namespaces and the exclusion of DTDs. Tim Bray, editor of the original XML 1.0 Technical Recommendation, recently introduced what he called a "thought experiment" – a "skunk works" effort to describe a new version of XML. While XML-SW (XML Skunk Works) isn't endorsed or supported by the W3C, it describes a new version of XML that seems very appealing. Tim Bray summarized the capabilities of XML-SW using a simple equation:

Author Bio

John Evdemon has been involved with XML since its meager beginnings.and has been a CTO at both large and small XML vendors. Currently the CTO of DiscoveryLogic (www.discoverylogic.com), John also serves as coeditor-in-chief of XML-Journal *and is an Invited Expert with the W3C XML Core Working Group.*

"For users, vendor compliance with open standards helps ease integration initiatives"

XML-SW = XML 1.0 (2nd Edition) – DTDs + Namespaces + XML:Base + XML Infoset

Folding Namespace, Infoset, and XML Base support into the XML specification consolidates the "core" XML specifications. Removing the guidelines for using DTDs avoids tying XML to a single schema language. Readers of *XML-J* are strongly encouraged to review XML-SW.

Web Services Standards

While Web services doesn't specifically require the use of XML, current trends recommend the use of WSDL, an IDL-like language for describing the capabilities of a particular Web service. SOAP provides the simple message handling and protocol requirements, while UDDI can provide a repository for registering, querying, and finding a Web service. Web services, like XML itself, ran the risk of potentially collapsing under its own considerable marketing weight. Several vendors provide support for Web services using SOAP, WSDL, and UDDI. Many of these same vendors, in an effort to gain market advantage, introduced proprietary flow languages and additional capabilities that were incompatible across disparate vendor implementations, making Web services aggregation efforts a formidable task. Vendors soon recognized that splintering Web services into multiple, incompatible initiatives placed the future adoption of Web services at risk.

To avoid this risk, the Web Services Interoperability Organization was organized. WS-I unites a number of large vendors (IBM, Microsoft, BEA, and others) to ensure cross-vendor compatibility, define Web services implementation models, and provide general guidance for the future of the Web services initiative. Although WS-I began as a vendor-sponsored initiative, several large Web services users are also represented (Ford, Qwest, and others). The goals of the WS-I organization are admirable and should help ensure wider adoption of Web services. Organizations interested in developing or deploying Web services should monitor the activities of WS-I closely; the group is expected to provide several best practices and tools to help ease the implementation of Web services.

Where Do We Go from Here?

The realization of XML as a lightweight messaging/integration language helped enable loosely coupled, Web-based integration platforms. The promotion and subsequent adoption of open standards have had both positive user effects and negative vendor effects. For users, vendor compliance with open standards helps ease integration initiatives. For some vendors, compliance with open standards has resulted in a potential commoditization of their platform, forcing them to expand their capabilities into the integration platform space. This expansion will result in an eventual convergence in the integration space (although the effect open standards will have on the integration platform space has yet to be determined). Regardless of the platform used, many integration initiatives will rely on XML-based messages, ensuring the broadest possible base of message consumers.

XML also continues to evolve, with the publication of the XML 1.1 Technical Recommendation expected late this year/early next year. Web services are also maturing; the WS-I initiative hopes to unite disparate vendor implementations and ensure the viability of Web services for large organizations.

The activities summarized in this article are positive developments for both integration initiatives and XML itself, ensuring the long-term viability of both technologies. In terms of EAI and B2B initiatives, we're rapidly moving toward standards-based, loosely coupled infrastructures – great news for users, but bad news for vendors (since yet another product line faces commoditization). For XML syntax, specification revisions ensure broader acceptance and better support for foreign languages via newer Unicode standards. With WS-I, Web services vendors and users are working together to ensure compatibility and recommend best practices for design and implementation. With regard to XML messaging formats, I'm once again reminded of incompatible power outlets.

Several XML messaging initiatives are under way, each of which appears to be defining its own version of "standard" e-business transactions. X12, having defined the only true e-business standards in use throughout America, seems to be abandoning its standardized transactions for a new, backwards-incompatible, core component–based design (heavily influenced by ebXML).

Will we ever realize a single set of XML schemas in which all participants define transactions in a similar manner? Asking several people will likely yield one of three answers:

1. No. It's simply not possible.
2. Yes. Come take a look at standard "X". We've been working on it for the past three years and hope to have something deployable within the next two.
3. This already exists. It's called EDI. Use an XML representation of EDI to ensure compatibility, deploy it, and move on to more interesting work.

Resources

- *J2EE Connector Architecture (JCA):* http://java.sun.com/j2ee/connector/
- *SOAP:* www.w3.org/2002/ws/
- *XML-RPC:* www.xml-rpc.org/
- *OAGIS (Open Applications Group Integration Specification) – fairly mature schema representing commonly used business transactions:* www.open-applications.org
- *RosettaNet – e-business language initially designed for electronics industry but expanding into other verticals:* www.rosettanet.org
- *xCBL (Common Business Library) – markup language used by Commerce One for its B2B integration products:* www.xcbl.org
- *UBL – supported by OASIS Technical Committee to extend xCBL to support additional, commonly used business transactions:* www.oasis-open.org/committees/ubl/
- *VCML (Value Chain Markup Language) – extends standard e-business transactions for several industries into XML syntax:* www.vcml.net
- *X12 (Accredited Standards Committee [ASC] X12) – defines EDI standards for U.S.:* www.x12.org
- *XML 1.1 Working Draft:* www.w3.org/TR/xml11/
- *XML-SW (Skunk Works):* www.textuality.com/xml/xmlSW.html
- *Web Services Interoperability Organization (WS-I) – designed to unite vendor initiatives and provide tools, guidance, and best practices for vendors and users deploying Web services:* www.ws-i.org/

JEVDEMON@SYS-CON.COM

XML JOURNAL The World's Leading XML Resource

Home | Advertise | Subscribe | Contact

Dot.com
- buyer's guide
- xml forums
- mailing list
- xml jobs
- xml store

Magazine
- advertise
- authors
- customer service
- editorial board
- subscribe

Content
- archives
- digital edition
- editorial
- features
- interviews
- product reviews
- source code

Conferences
- xml edge
santa clara, ca
oct 22–25

Search XML-J
Search

Check out these companies and their committed XML technology!

XML-J Specials

JDJStore.com
$79.00

Bestsellers
1. JBuilder Enterprise Upgrade 6.0 From JBuilder 6.0 Enterprise $1799.99
2. Borland JBuilder Pro V6.0 Single $849.99
3. JBuilder Enterprise Upgrade 6.0 From any previous Enterprise Edition $2249.99
4. JBuilder 6 Personal 98/2K/NT/Linux/Solaris $53.99
5. WebGain Visual Cafe 4.5.2 Enterprise Suite $2749.99

What's Online

www.sys-con.com/xml

Readers' Choice Awards

What products do you trust most to get your work done? Starting July 1, you can vote online for your favorites in the 2002 *XML-Journal* Readers' Choice Awards.

This year's survey promises to be bigger and better than ever, with an expanded list of product categories and exciting additions.

Winners will be announced at the **XMLEdge 2002 West–International Conference & Expo** in October.

XMLEdge 2002 West

Be a part of the *i*-technology event of the year. Online registration starts July 1 for the **XMLEdge 2002 West–International Conference & Expo,** to be held at the San Jose McEnery Convention Center in San Jose, California, September 30–October 3. This year will feature top-level XML professionals in the industry as well as an Expo twice the size of last year's show.

Free Weekly Newsletters

SYS-CON's industry newsletters are informative, convenient, up to the minute – and absolutely **FREE**. Just look in your e-mail inbox every week. Learn what's hot and what's not at your leisure, because these newsletters can be read anywhere you have access to a computer. Subscribe to ***XML-J* Industry Newsletter**, and check out the other **SYS-CON**-family newsletters while you're at it for news on Java, Web services, wireless, Linux, and ColdFusion. Sign up at www.sys-con.com!

Subscribe to Our FREE Weekly Newsletters
- ☑ Java
- ☑ XML
- ☑ Web Services
- ☑ Wireless
- ☑ ColdFusion
- ☑ WebSphere
- ☑ WebLogic
- ☑ Linux
- ☑ .NET

Conference Newsletters
- ☑ JDJEdge
- ☑ XMLEdge
- ☑ Web ServicesEdge
- ☑ WirelessEdge

E-Mail Address:

Subscribe

XML Store Resource CD Special

For a limited time you can purchase the ***XML-J*** and ***WSJ* Resource CD** at $40 off the regular price. This is a complete library of articles on one CD, with over 40 chapters and 400 articles you can't find anywhere – except by visiting www.xml-journal.com.

XMLJList

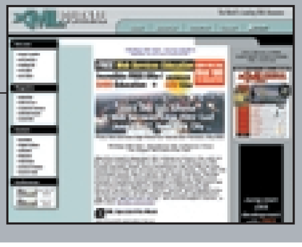
All-new from **SYS-CON Media** is the **XMLJList**, a **FREE** service that allows you to connect with other XML professionals, take part in XML discussions, ask technical questions, and more. As a List member, you also have access to the **XMLJList** archives, a valuable source of information. Go to www.sys-con.com/xml/list.cfm

Write for Us

Do you have something you want to write about the XML world? We're interested in real-world implementations, industry analysis, and tutorials. Check out the author guidelines at www.sys-con.com/xml/writers and send us a proposal.

XML-Journal.com

The most complete resource for XML news, events, products, and job searches is at www.xml-journal.com. Get a leg up on the competition by learning what's happening, minute by minute, in the IT world.



QUICK POLL
Who should receive most credit for the ever-rising importance and status of XML today?
- ○ A- Charles F. Goldfarb
- ○ B- Tim Bray
- ○ C- Jon Bosak
- ○ D- W3C
- ○ E- Microsoft Corp.

Vote
Results

xml spy FREE DOWNLOAD

Sprint PCS® Application Developer's Conference ▸ August 25-28 Register now!

VoiceGenie VoiceXML

Click for a FREE 30-day trial AltoWeb

Sprint PCS® Application Developer's Conference ▸ August 25-28 Register now!

XML NEWS

ProActivity Update Has XML Export Capability

(Newton, MA) – ProActivity 4.0, a platform for capturing, analyzing, sharing, and optimizing business process knowledge across the enterprise, has been released by ProActivity, Inc.

In addition to the ability to automate the requirements definition and logical design phase of the system development life cycle (SDLC), ProActivity 4.0 has a new XML export capability, ProActivity Process eXchange (PAPX), and support for DB2, IBM's relational database management system.

www.proactivityinc.com

Pageflex Adds Real-Time Document Editing Tool

(Cambridge, MA) – Pageflex, Inc., has expanded its publishing product line with the addition of .EDIT. The tool enables end users to use their Web browsers to remotely select, edit, and style text, images, and shapes without having to install software; create and position new document elements such as text boxes, images, and shapes; and create documents that use corporate-approved fonts, colors, design elements, and images.

www.pageflexinc.com

New Features for Altio Platform

(Boston) – Version 2.5 of the AltioLive platform now features support for Web services, enhanced graphing capabilities, and improved reporting functions as well as new drill-down and tree-view options. AltioLive 2.5 also offers enhanced JDBC drivers for improved database access.

www.altio.com

Software AG Offers XML-Based Insurance Solution

(Reston, VA) – Software AG, Inc., has introduced SAGe Insurance, an XML-based insurance solution designed to reduce IT expenses and time required to implement ACORD XML standards for exchanging information among institutions, sales channels, and trading partners.

The new product combines Software AG expertise – hence *SAGe* – and the EntireX XML Mediator product to create, transform, and sequence ACORD XML documents in a real-world dynamic environment. EntireX XML Mediator provides the capability to receive ACORD XML documents from any source, using a complex rules base to determine how each document should be processed and translated into a set of XML and non-XML messages. The documents are then routed in their entirety, or in relevant sections as defined by the organization's business rules, to appropriate internal or external systems.

www.softwareagusa.com

Factiva Launches Web API Based on XML

(New York) – Factiva, a Dow Jones & Reuters Company and global news and business information provider, has a new application programming interface that allows developers to create customized applications that integrate Factiva's content and functionality into end users' workflow systems.

The XML-based API provides developers with maximum flexibility and a simplified development path to customization.

www.factiva.com/developerkit

Document Conversion System New from Exegenix

(Barcelona) – Exegenix has unveiled a technology that transforms an organization's new and legacy content into XML.

Exegenix solves the problems of content conversion by interpreting a document's logical structure based on the appearance and position of its components. This approach allows any PostScript or PDF document, regardless of its complexity, to be quickly and cost-effectively transformed into XML.

www.exegenix.com

Ipedo Offers Web Express XML Framework

(Redwood City, CA) – Ipedo has introduced Ipedo Web Express, a framework developers can use to leverage XML in their Web sites and portals. Reportedly the first product of its kind, Web Express allows organizations to reduce the complexity of site style changes, reuse existing content assets in new applications, and streamline development for new dynamic Web applications.

Features include modular content processing, simple tag execution, cross-document content combination, and an open development environment.

www.ipedo.com

Adobe Upgrades FrameMaker Product

(San Jose, CA) – The newest version of Adobe Systems' XML authoring and publishing solution is now on the market.

FrameMaker 7.0 includes the structured authoring tools for both XML and SGML (previously available as a separate purchase) as well as a WYSIWYG word processing interface. In addition, FrameMaker now supports World Wide Web Distributed Authoring and Versioning (WebDAV) and Adobe's XMP (Extensible Metadata Platform).

www.adobe.com

Avaltus Launches Jupiter Suite 4.5

(Denver) – Avaltus, a global provider of e-learning software and services, is now shipping Jupiter Suite 4.5, a 100% Java/XML-based Learning Content Management System (LCMS) that unifies learning content across the enterprise into a seamless data repository. This allows organizations to convert information currently stagnating in corporate networks, libraries, and media files into interactive and reusable courseware.

Some of the new features: a scalable, multitier architecture; one-stop loading of future upgrades; controlled versioning; and improved GUI.

www.avaltus.com

New OASIS Standards Approved

(Boston) – Three OASIS Committee Specifications – Directory Services Markup Language (DSML) v2, ebXML Registry Service (RS) v2, and ebXML Registry Information Model (RIM) v2 – are now officially OASIS standards, having received final approval from the OASIS membership at large.

To reach this milestone, each has been approved by its respective OASIS technical committees, implemented by a minimum of three organizations, completed a 90-day review, and been submitted to a final vote by the full OASIS membership.

Members of the ebXML Registry Technical Committee include Boeing, Intel, IONA, Logistics Management Institute, NIST, Sterling Commerce, Sun Microsystems, and Vitria Technology. Members of the DSML Technical Committee include Access360 and Novell.

www.oasis-open.org

XML NEWS

Zero G Updates InstallAnywhere

(San Francisco) – Zero G Software's latest version (5) of its InstallAnywhere product has enhanced XML support, suite installation capabilities, merge module support, new team development features, Web services–based application deployment tools, drag-and-drop functionality, and Apache Ant integration.

www.ZeroG.com

Choicelinx, Voice Channel Team Up for New Product

(Manchester, NH / Boston) – Choicelinx Corporation, a benefits technology company specializing in customer-designed benefits products for the health care industry, and Voice Channel Corporation, developer of voice Web applications for human resource and health care organizations, have agreed to integrate the benefits management and enrollment technology from Choicelinx with Voice Channel's benefit information system, a voice Web product.

The Choicelinx integrated suite of software applications enables health insurers to offer Web-based enrollment, plan selection, and consumer-driven plan offerings to their customers. Integration with the Voice Channel technology will create a tool that gives consumers the option of managing their entire benefits process using the Internet or a telephone interface with real-time speech recognition.

www.choicelinx.com
www.voicechannelcorp.com

ACOM Adds Flexible Merge Feature to EZeDocs

(Duluth, GA / Long Beach, CA) – The Duluth-based iSeries/400-AS/400 Division of ACOM Solutions, Inc., has enhanced version 5.1 of its EZeDocs document output management solution with a data merge feature that captures and inserts additional data dynamically into documents created from spool files using electronic form templates.

EZeDocs allows companies to replace conventional preprinted business forms with customized electronic templates and enables the delivery of completed documents as laser-printed hard copies, e-mail, fax, and PDF files for attachment or archiving. It also allows iSeries-AS/400 users to route internal and external electronic documents intelligently, based on data, as well as to access the worldwide communications infrastructure directly from the computing platform.

The flexible merge feature is available with new installations of EZeDocs version 5.1. Current users under maintenance contracts can download it free from www.acom.com.

Altova Releases XML Spy 4.4 Suite

(Beverly, MA) – XML Spy 4.4 Suite, a comprehensive product line of developer tools for advanced XML application development consisting of the XML Spy 4.4 IDE, the XML Spy 4.4 XSLT Designer, and the XML Spy 4.4 Document Editor, is now available from Altova.

New features include DocBook editing support, enhanced Web services standards conformance in the SOAP client and debugger, a built-in multilanguage spell-checker, and separate English language medical and legal dictionaries.

The suite can be downloaded and purchased from the XML Spy Online Shop, www.xmlspy.com/order.

LogiXML Expands Products to Microsoft .NET Platform

(McLean, VA) – LogiXML, Inc., creator and designer of the XML-based modeling language LogiXML, has announced that the language is now available on the Microsoft .NET platform.

In performance evaluations the new .NET version of LogiXML has shown improvements in scalability – up to 400% improvement over earlier versions. The addition of the .NET framework, along with a number of other changes, has also shortened integration time by providing a larger number of Web systems that support Web services and protocols through SOAP.

www.logixml.com

New XML-Based Platform from Enosys Software

(Redwood City, CA) – Enosys Software has introduced the Enosys XQuery-based suite of products, the industry's first XML-based platform to enable real-time information integration inside and outside company firewalls.

The Enosys platform is an advanced Enterprise Information Integration (EII) solution based on open Java standards and patent-pending XQuery technology. It consists of an integration server, a design suite, and management tools. Key features are uniform access to the full variety of data sources; integration and presentation in a single, optimized data layer and delivery of the data back to requesting Web applications and services in real time; and simple querying capability.

www.enosyssoftware.com

Course Technology and Altova Help Students Tackle XML

(Boston) – Course Technology, an IT publisher and part of The Thomson Corporation, will bundle Altova's XML Spy 4 software suite with its XML books and courseware, enabling students to have a fully integrated learning experience.

www.thomson.com
www.xmlspy.com

SilverStream Offers eXtend QuickStart for RosettaNet

(Billerica, MA) – SilverStream Software has announced SilverStream eXtend QuickStart for RosettaNet, a solution for RosettaNet Basics that enables organizations to implement dynamic, global trading networks built on established RosettaNet standards.

SilverStream eXtend features breakthrough technology for XML integration to legacy systems, business process management, interactive applications, personalization, content management, user management, wireless device support, and more.

www.rosettanet.org
www.silverstream.com

XCache Technologies Ships XCompress 1.0

(Bellingham, WA) – XCache Technologies is now shipping XCompress 1.0. XCompress adds value to static and dynamic Web applications by compressing outgoing text between the Web server and the client. Page response times improve by a factor of three or more while overall bandwidth use can drop by two thirds or more.

www.xcache.com

Inmagic's DB/Text WebPublisher 6.0 Released

(Woburn, MA) – Inmagic, Inc., a global provider of content and information management software and services, has announced the release of DB/Text WebPublisher 6.0, with XML support.

www.inmagic.com

Rube Goldberg Would Have Loved XML Schema

What good is a specification if no one can properly interpret it?

BY **JOHN EVDEMON**
Coeditor-in-Chief, XML-Journal

Rube Goldberg, Pulitzer Prize–winning cartoonist who illustrated complex ways to achieve easy results, saw his cartoons as "symbols of man's capacity for exerting maximum effort to accomplish minimal results." He believed there were two ways to do things: simple and hard, and that a surprising number of people preferred the latter.

At this writing the XML-Dev mailing list is debating the role of XML for IETF protocols (more specifically, the requirement to use XML Schema when defining XML-based IETF protocols). (XML-Dev is a high-traffic mailing list targeting XML developers. Readers are encouraged to join and participate at www.xml.org/xml/xmldev.shtml).

Why the concern? XML Schema has been a W3C Technical Recommendation for well over a year. You'd expect the members of XML-Dev (many of whom are well-known thought leaders in the XML community) to endorse the use of XML-based IETF protocols. While the group seems to endorse the concept of XML-based protocols, controversy arises when these protocols must use XML Schema to define the elements, attributes, structures, and datatypes used by the protocol. Depending on whom you talk to, XML Schema may be regarded as overly complicated or a panacea for supporting both simple and complex datatypes.

The source of the controversy is a new (6/4/2002) Internet Draft, "Guidelines for the Use of XML within IETF Protocols" (downloadable from http://search.ietf.org/internet-drafts/draft-hollenbeck-ietf-xml-guidelines-04.txt), that describes basic XML concepts, analyzes alternatives in the use of XML, and provides guidelines for using XML within IETF standards-track protocols.

While XML wasn't originally designed to be a protocol, its ability to model complex structures in an open manner lends itself to such. A quick glance at recent Internet Drafts reveals over 40 separate initiatives using XML. It became fairly obvious that some sort of "best practices" for designing XML-based protocols would be needed. Section 4.6 the Internet Draft recommends that XML Schema be adopted as the standard formal mechanism to define structural and data content constraints for XML protocols. Specifically:

> *Protocols should use an appropriate formalism for defining validity of XML protocol elements. XML Schema should be used as the formalism in the absence of clearly stated reasons to choose another.*

This seemingly harmless statement has caused a bit of an uproar. Miloslav Nic, founder of Zvon (www.zvon.org), states flatly: "It would be a disaster if IETF adopted XML Schema." An XML Schema expert, having designed and built the free XML Schema tutorials on the Zvon site, he readily admits that each time he approaches XML Schema he feels "like a novice." Jiri Jirat, coauthor of the tutorials, also admits having "difficulty." Both men are quite familiar with XML and XML-related technologies, having developed over 15 in-depth tutorials for Zvon.org. If these guys are having trouble understanding the spec, what hope does an average developer or group of analysts have? As they say, "We're not aware of anyone who would claim that he or she really understands the specification."

The XML Schema specification is overly ambiguous, enabling developers to interpret its various aspects in radically different ways. According to Miloslav, "Our interpretation can change several times during a single discussion." Specifications are supposed to cut through the clutter by clearly defining the terms, concepts, syntax, and rules for working with a particular technology. XML Schema seems to have failed us in this area.

Realizing the intricate nature of the spec, the designers produced an XML Schema Primer. While useful, it fails to clearly illustrate the more arcane portions of the overall specification. Understanding the primer is clearly not enough. "If you get away from it for a week or two, it's like starting from the beginning," says Miloslav. "Adopting XML Schema for future standards development could create tremendous problems for everyone."

Not everyone feels this way. According to JP Morgenthal (coeditor-in-chief of ***XML-Journal***), "While it's true that XML is designed to be simple, there's no reason that concepts, tools, and applications built using XML need to be implicitly simple." In JP's opinion, "exchanging data across stove-piped applications and different organizations requires a complex representation that best allows organizations to ensure equality and cleanliness of the data."

While JP raises some interesting points, nothing prevents a comparatively simple solution from solving a rather complex task. Indeed, a primary reason for the popularity of Web services is that they're relatively easy to understand and implement. RELAX NG proves that difficult validation requirements can be met using fairly simple (i.e., elegant) designs. RELAX NG is capable of performing much of the functionality of XML Schema (and more) using a compact, easy-to-understand specification. The more mature XML Schema, however, has been endorsed by the W3C (which some rather short-sighted managers feel should be a requirement). The XML community must avoid painting itself (or being painted) into a corner with XML Schema. Other, possibly better, alternatives are available – each should be explored in its own right.

What do you think? Debate the merits of XML Schema, RELAX NG, Schematron, et al. in the *XML-Journal* Forums at www.sys-con.com/fusetalk/categories.cfm?catid=4.

JEVDEMON@SYS-CON.COM

ADOS Co., Ltd.

http://www.a-dos.com

Altova

www.xmlspy.com